Active Control for Lightweight Autonomous Tricycles

Iñigo Martínez López

Submitted to the School of Engineering Tecnun - University of Navarra in partial fulfillment of the requirements for the degree of Master of Science in Industrial Engineering

Master thesis carried out at the Massachusetts Institute of Technology , Media Lab - Changing Places Group

June 2017

Active Control for Lightweight Autonomous Tricycles

Iñigo Martínez López

Submitted to the School of Engineering Tecnun - University of Navarra
on May 20, 2017, in partial fulfillment of the requirements for
the degree of Master of Science in Industrial Engineering

Abstract

Most of the trips in the cities are one-person, low-speed, short distance. That is why Changing Places group at the MIT Media Lab has designed an autonomous, electric and shared urban vehicle that will take conventional cars out from urban areas. This vehicle concept is called Persuasive Electric Vehicle (PEV). An agile, on-demand, shared and functionally-hybrid tricycle, which is thought to minimize traffic congestion, energy consumption and pollutant emission. The PEV takes advantage of existing bicycle lanes, provides energy-efficient mobility, and addresses sedentary lifestyles.

One meter wide, the PEV is also know as a narrow track vehicle. Due to its dimensions, roll stability is a real issue. The PEV would have to lean into corners in order to compensate for the lateral acceleration and maintain their stability. Therefore, this thesis has been dedicated to the design and fabrication of a three wheeler vehicle with an active tilting system. The proposed tilting strategy would be automatic and achieved by a dedicated tilting actuator, requiring no additional inputs from the driver.

First, the dynamic model of a tilting three wheeler vehicle is replicated, from which a linear model is derived. Then the output feedback regulator is designed, by means of optimal control strategies (Linear-Quadratic Regulator), built to optimize the lateral acceleration of the PEV. This controller has the longitudinal velocity as a parameter, and leads to the minimization of the tilting torque and of the lateral acceleration perceived by the driver, and have good performances as well as good robustness properties.

Key words: *tilting vehicle; narrow vehicle, vehicle dynamics, modeling, three-wheeler, lateral stability, cornering behaviour, urban vehicle, direct tilt control, robust control, H_2 control, gain scheduled controller*

Thesis Supervisor: Luis Matey Muñoz
Title: Associate Professor

Acknowledgments

Thank you *Kent Larson*, for your mentorship, input, and advice over the last year. I feel that my time in the Changing Places Group has allowed me to grow as both a designer and engineer. Thank you for taking me into the your group and giving me the opportunity to be part of the MIT Media Lab.

Thank you *Michael Lin*, for sharing your excitement around this area of research and your thoughts with me based on a tremendous amount of experience and knowledge.

Thank you *Luis Alonso*, for challenging me and bringing the best in me.
Your gentle and patient guidance has allowed these ideas and projects to flourish.

Thank you *Arnaud Grignard* for being a confidant throughout this process. Big brother.

Thank you *Lucas Cassiano*, for always taking the time to discuss ideas no matter how big or small, and for offering your generous support.

Thank you *Changing Places people* for being a supportive second family.
We have shared many good times and ideas.

Thanks to everyone that, one way or the other, played a part in this story: fab-lab managers, facilities, academic officers...

Thanks to the *Media Lab community* for being as wild and brilliant as you are.

Thanks to *my family* for the support and love always, even when it is tough.

Thank you *Naroa*, for being my other half in everything.
We are one in the same since the beginning.

Contents

List of Figures

List of Tables

Acronyms

NTV Narrow Tilting Vehicle

DOF Degree(s) Of Freedom

DTC Direct Tilt Control

STC Steering Tilt Control

SDTC Steering and Direct Tilt Control

PID Proportional Integral Derivative

LQR Linear Quadratic Regulator

Notations

ψ	Vehicle yaw angle
θ	Vehicle tilt angle
y	Lateral vehicle distance from instantaneous center of circular path
m	Mass of the vehicle
m	Mass of the vehicle without wheels
m_f	Mass of each front wheel
m_r	Mass of the rear wheel
h	Height of center of gravity (c.g.) of vehicle from ground
h_f	Height c.g. of front wheels from ground
h_r	Height of c.g. of the rear wheel from ground
h_o	Height of c.g. of vehicle without wheels from ground
L_f	Longitudinal distance from center of gravity to front axle
L_r	Longitudinal distance from center of gravity to rear axle
L_r	Longitudinal distance from c.g. of vehicle to c.g. of vehicle without wheels
I_x	Tilting moment of inertia of the vehicle
I_z	Yaw moment of inertia of the vehicle
$I_z\,o$	Yaw moment of inertia of the vehicle without wheels
$I_{wheel\,f\,\theta}$	Tilting inertia of each front wheel about its own axis
$I_{wheel\,f\,rot}$	Inertia of each front wheel about its rotating axis
$I_{wheel\,f\,\phi}$	Yaw inertia of each front wheel about its own axis
$I_{wheel\,r\,\theta}$	Tilting inertia of the rear wheel about its own axis
$I_{wheel\,r\,rot}$	Inertia of the rear wheel about its rotating axis
$I_{wheel\,r\,\phi}$	Yaw inertia of the rear wheel about its own axis
R_{wf}	Front wheel radius
R_{wr}	Rear wheel radius
F_{zf}	Vertical reaction in each front wheel
F_{zr}	Vertical reaction in the rear wheel

C_f	Cornering stiffness of each front wheel
C_r	Cornering stiffness of the rear wheel
λ_f	Camber stiffness of each front wheel
λ_r	Camber stiffness of the rear wheel
δ	Front wheel steering angle
β	Front wheel trail angle
γ	Front wheel trail
$\gamma_{pneumatic}$	Front wheel pneumatic trail
M_δ	Steering torque input from driver
M_{trail}	Camber stiffness of the rear wheel
M_t	Tilt torque from actuator
α_f	Slip angle at front wheel
α_r	Slip angle at front wheel
V_x	Longitudinal vehicle velocity
R	Curvature radius of the curve path
b	Width of the vehicle in the base

o

1. Introduction

From Drive-less Cars to Car-less Cities

The challenge of moving people and goods around the world's dense, growing major cities is hard and getting worse. Nearly ninety per cent of the world's population growth will be concentrated in Asia and Africa, and is expected that 2.5 billion people will add to the world's urban population by 2050. Furthermore, cities will soon account for 80 percent of global carbon emissions, according to the United Nations urbanization prospects.[1]

[1] Department of Economical United Nations and Social Affairs. World urbanization prospects: The 2014 revision, highlights, 2014. ISBN: 978-92-1-151517-6. Available at http://esa.un.org/unpd/wup/Highlights/WUP2014-Highlights.pdf

Much of that emissions will come from idling cars stuck in kilometers-long traffic jams. Cities around the world are striving to improve livability by way of reducing dependency on fossil-fuel vehicles, applying controversial policies and restrictions, usually not understood by the citizens. Hence, for the moment no key solution has been found. In fact, this problem is much more complex and will require long-term solutions.

In this way, technology is evolving to address better urban transportation, more secure and environmentally friendly urban transportation. In the coming decades, the transportation world will be totally transformed by three concepts: **Electrification, autonomy and shared economy.**

Electrification

The immediate way of getting rid of the pollution that internal combustion engines produce is to install electric motor based drive-trains in our cars, which account for the most percentage of the emissions.

The introduction of a fully electric transportation system will need new infrastructure, charging stations and more capacity in the electrical grid. However, it is undeniable that with the exhaustion of fossil fuels, electric vehicles will replace current cars in the near future. The appearance of hybrid propulsion systems supposes the first step towards the electrification. While it is clear that at the moment electric vehicles present some limitations, such as battery range, the development of energy storage technologies will get over these issues in the next years.

Autonomy

Nowadays, self-driving vehicles are almost a reality, thanks to the improvements in artificial intelligence and computer technologies.

On one side, the development of new techniques and algorithms in artificial intelligence have expanded the capabilities of conventional cars. Deep learning techniques, along with artificial neural networks have been able to achieve high levels of autonomy by using data coming from different sensors – *cameras, ultrasonic, radar and lidar* – that give enough information about the car's surroundings.

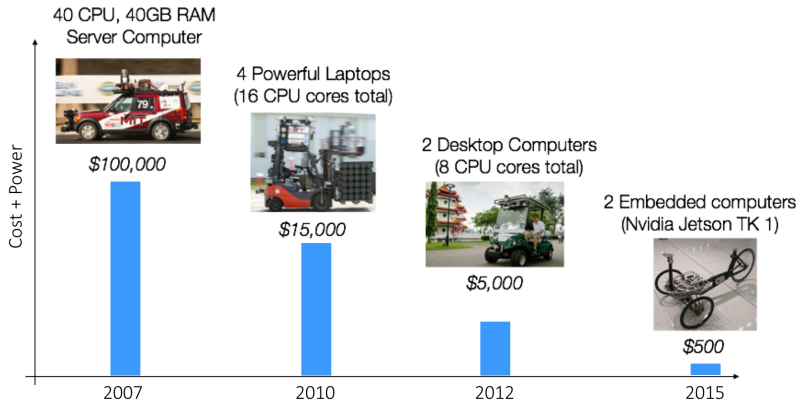


Figure 1: The cost and the computational power required for self-driving cars has been greatly reduced during the last decade.

On the other side, throughout the last decade there has been an incredible decrease in the cost of computation time and in the size and power to process the data coming from the sensors. This point is critical in order to incorporate compact and affordable autonomy packages in current cars.

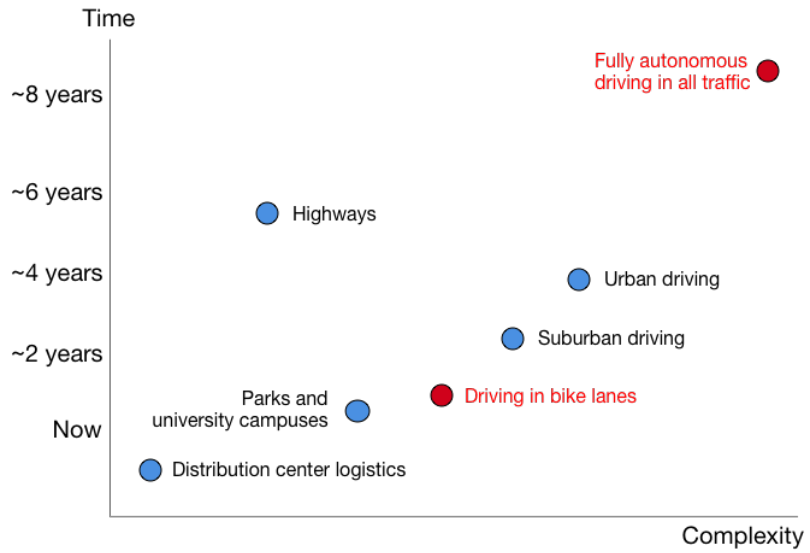


Figure 2: Challenges of self-driving vehicles based on the time of deployment and the complexity of the task.

However, achieving a level 5 of autonomy (based on SAE International Standard J3016[2]) will require years of development, due to the complexity of the task. Driving in bike lanes, on the contrary, makes up a straightforward challenge, basically because the road conditions and the sensors required for this task are not that demanding. The use of less and more affordable sensors will allow the deployment of intelligent three –or two in the future– wheeler vehicles through bike lanes.

[2] Society of Automotive Engineers (SAE). Taxonomy and definitions for terms related to on-road motor vehicle automated driving systems, 2014. Available at http://standards.sae.org/j3016_201401/. Summary available at https://www.sae.org/misc/pdfs/automated_driving.pdf

Sharing Economy

During the last couple of years, the sharing economy has disturbed the traditional way of moving in cities. Companies as *Uber, Lift, BlaBlaCar, GetAround, Turo, Zipcar* and many others have offered a new way of commuting in urban areas. In this new paradigm, the technology has arrived to some countries without a proper regulation, which has caused complaints and protests from conventional transit services. Other sectors, as housing (*Airbnb*), goods (*Wallapop*), food (*LeftoverSwap*), sports equipment rentals (*Spinlister*), and tourism (*Vayable*) have also seen the irruption of this new way of economy.

More and more cities around the globe have adopted so called bike-sharing systems, which enable citizens and visitors to pick up bicycles at some point in the city, use them and leave them at another point in exchange for a small fee.

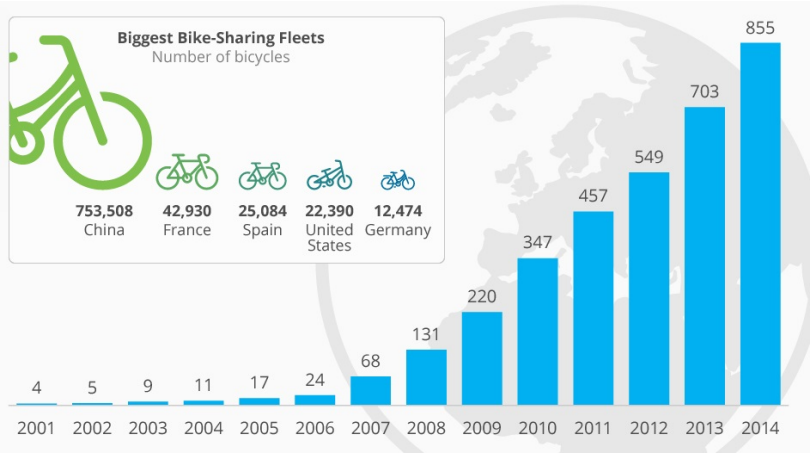


Figure 3: Number of cities worldwide that offer bike-sharing services. Source: https://www.statista.com/chart/3325/bike-sharing-systems-worldwide/

Car-less Cities

The ultimate policy will be to get rid of the vehicles in the city centers, as multiple cities have already announced.[3] These can be seem as very radical strategies in transport policies, but the congestion and the pollution big cities are suffering at this moment exhibit much more issues and harm.

Therefore, is it possible to leverage electrification, autonomy and the shared economy to help fulfill this vision while ensuring a convenient flow of people and goods across the city?

[3] Leanna Garfield (Business Insider). 12 major cities that are starting to go car-free, 2017. Available at http://www.businessinsider.com/cities-going-car-free-2017-2

Persuasive Electric Vehicle

Changing Places group[4] at MIT Media Lab[5] believes in the progressive fading of conventional cars from urban areas. The group's vision recalls on Kent Larson's[6] ideas, who affirms that the introduction of autonomous, electric and shared vehicles will hugely impact the cities of the future: Different streets layout, no more parking problems, less necessary vehicles and the overall result will be that the citizens will take the streets back from the cars.

Furthermore, most trips in the city are one-person, low-speed, short distance. It does not make any sense to put one person in a 1800 kilograms vehicle to move across a city only a couple of kilometers. It was in front of this situation when the Changing Places group designed an ultra-lightweight one-person, three-wheel vehicle that is bike-like, not car-like.

[4] Changing Places group projects. http://cp.media.mit.edu/

[5] About the MIT Media Lab. https://www.media.mit.edu/about/mission-history/

[6] Changing Places group principal investigator Kent Larson. https://www.media.mit.edu/people/kll/overview/

Figure 4: Persuasive Electric Vehicle (september 2016)

The **Persuasive Electric Vehicle**[7] (PEV) is an agile, on-demand, shared and functionally-hybrid tricycle with an expected contribution of 60% emissions reduction and 30% vehicle-distance reduction. The PEV is thought to constitute a new and indispensable category of vehicles in the emerging constellation of mobility systems.

[7] PEV information. http://cp.media.mit.edu/pev/

The PEV takes advantage of existing bicycle lanes, provides energy-efficient mobility, and addresses sedentary lifestyles. Designed with a cover to protect from the rain and the option for electric assist, PEV makes biking compelling for various demographics.

Various **persuasive interventions** are displayed through user interaction with smartphones to facilitate pedaling behavior. Influential strategies are designed for both the interior and exterior of PEV. For example, an interior display shows how many previous riders have actually pedaled while riding a particular PEV. The exterior of the PEV changes color depending on whether a rider actually pedals or not.

Figure 5: Persuasive Electric Vehicle render model

With a fabric exterior shield, a foldable canopy, and a 250-watt assist motor, the autonomous tricycle is "persuasive" in that it is designed to "encourage positive modal shifts in mobility behavior in cities." It has a top speed of 12 miles per hour, limited by the US regulation. It can be adapted for a human rider or for package transport, and it has the sensors and intelligence to operate autonomously.

Functionality

– **In transportation of goods** mode, the PEV will cover delivery on what is known as "the last urban mile". The PEV will be used as an autonomous courier service taking advantage of its attributes –small and maneuverable– to efficiently service congested city areas even in cases of traffic cuts.

Figure 6: Logistics Mode PEV

It could become a pervasive mode for urban package delivery, moving easily through crowded streets and reducing congestion and carbon emissions from gas-powered delivery vehicles.

– **In passenger transport** mode, the PEV would function as a bicycle providing more safety and comfort thanks to its structural design, which offers more protection than conventional bikes, as well as its electrical assistance system, easing effort while pedaling.

Figure 7: Passenger Mode PEV

Opposed to most shared bike systems, the PEV would "redistribute itself". Traveling autonomously from the drop-off point of one passenger to its next user's location, thus solving the problem of shortages and oversupply of bikes at different times in different locations.

As for the **persuasive element**, studies show that those using ride sharing programs were already walking or riding their bikes more than others. Since the PEV is meant to expand the market for sustainable transportation, its focus is to give automobile drivers a reason to switch over-which is, a little exercise. By pedaling, riders generate energy that is stored up and used when needed by the motor. Passengers could choose to take care of their bodies, cities, and environment at the same time.

Concluding Remarks

The PEV is one of the upcoming new technological innovations that are set to change the way people commute in urban areas. Concerns over pollution and congestion of cities due to the increasing numbers of vehicles have seen man engage in inventions to improve their communities. Automobiles contribute largely to pollution; the development and use of the PEV would introduce an ecological mode of transportation. The elderly will also have reliable means of movement, which in turn encourage exercise and fitness.

Motivation

The development of a new concept of intelligent vehicle opens the door to new research projects, in a **wide range of fields**. Computer vision, vehicle fleet simulations, persuasive technologies, mechanical design...are only few examples. Due to the open research opportunities, there is the necessity to understand in first place what people feel about the vehicle, from a user point of view. The conclusions of this analysis will narrow down the spectrum to this project's topic.

Amazon Products Reviews

A good way of understand the feelings of the people is to ask previous users about similar products. In this case, the most similar product was found in **adult tricycles**. The PEV's user demographics coincides with these vehicles users. Nowadays, one of the best databases of available user information can be found in Amazon[8].

[8] Amazon.com product example. https://www.amazon.com/Schwinn-Meridian-Adult-26-Inch-3-Wheel/dp/B01I92S3F4

However, this information is not easily accessible. While it is true that Amazon used to provide access to product reviews through their Product Advertising API to developers and sellers, the company discontinued that on November 2010, preventing customers from displaying Amazon reviews about their products, embedded in their websites. As of now, Amazon only returns a link to the review.

Therefore, getting this data requires other methods. In this project, the dataset of Amazon reviews was downloaded using a **web crawler on the HTML** pages[9]. Given a first level domain ("com") and the list of IDs of Amazon products, this Perl script automatically downloads, from the Amazon server that is dedicated to that domain, all and only the HTML pages that contain the reviews about that products.

[9] Andrea Esuli. Perl scripts for amazon reviews downloader and parser. https://github.com/aesuli/Amazon-downloader

Another Perl script extracts all the reviews contained in the downloaded HTML files, outputting for each review a record with the following information:

- A counter of the extracted reviews
- Date of the review in YYYY/MM/DD format
- ID of the reviewed product
- Star rating assigned by the reviewer
- Count of "yes" helpfulness votes
- Count of total helpfulness votes ("yes"+"no")
- Date of the review
- ID of the author of the review
- Title of the review
- Content of the review

Figure 8: One of the selected Amazon products

The selected products are really similar, with color, some components and accessories as main distinctions. The Figure 8 illustrates an example of one of the selected products, and more information has been included in Table 2.

Product ID	Product Description	No. Reviews
B000IORU06	Schwinn Meridian Adult 26-Inch 3-Wheel Bike	401
B000Z89JFO	Kent Adult Westport Folding Tricycle	157
B003F1WMZC	Worksman Port-o-Trike Three Speed Adult Tricycle	42
B00AWNI22I	Schwinn Meridian Tricycle (26-Inch Wheels)	11
B00FXMIUPW	Mantis Adult's Tri-Rad Folding Trike	2
B00KDJC5UQ	Komodo Cycling 24, 6-speed Adult Tricycle	2
B00G3N5MZ6	Torker 24 x 20 TriStar 2.1 Adult Trike 3 Speed	1

Table 2: Detailed information about the analyzed Amazon products

Overall, over 670 reviews were extracted, being the Schwinn model the product with most of the reviews. Obtaining this information required a great deal of time, since Amazon does not like to have its web-page scraped, and the access gets banned, so a delay has to be introduced. This sleep delay starts with 5 seconds and increases over time, making really tedious to get the review data.

An alternative to this method would have been to use an already existing dataset. This option was studied, but it was not able to find any of the scoped products on the dataset, due to the available period of data (up to July 2014).

Having extracted the data, different approaches were implemented in parallel to understand the overall opinion of the users. In this way, it has to be pointed out that the review comments were predominantly too brief and that hardly expressed valuable ideas or details about the product. The majority of the users just valued the product from a very general perspective. Even with more than 670 reviews, the quality of the reviews could mislead the data analysis.

In the next section three analysis are explained, with their corresponding scripts included in the Appendices. I would encourage the reader to review the scripts after the following section, since the scripts include much more information and details while here brief explanations and the main conclusions are going to be explained.

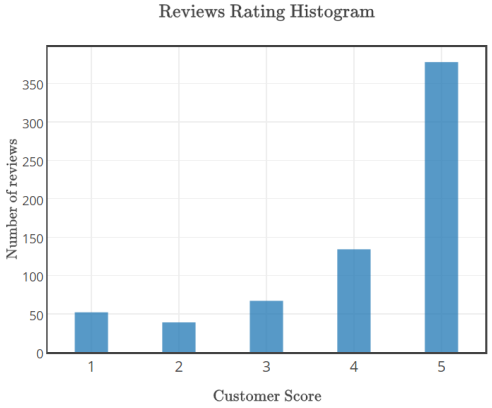


Figure 9: Reviews Rating Histogram

Sentiment Prediction Analysis

The purpose of this analysis was to investigate the positive and negative attitudes towards the selected products. Sentiment analysis attempts to determine which features of text are indicative of it's context (positive, negative, objective, subjective, etc.) and build systems to take advantage of these features.

Generally speaking, a satisfied customer will give 5 stars to the product, whereas few users will give poor evaluations (only if something really bad happened). This behavior is visualized in Figure 9, where the 5-star evaluation prevails.

This skewed distribution makes difficult to do a sentiment analysis based on the star-evaluation of the user. That is why the focus was not on the Score, but only in the positive/negative sentiment of the recommendation. Out of this analysis, the most positive and negative words are obtained:

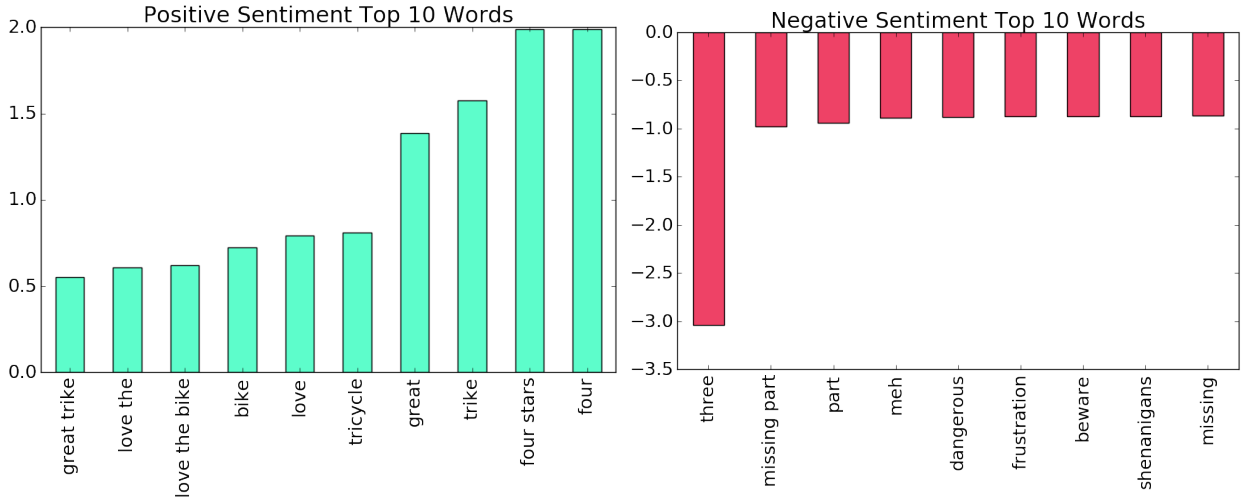


Figure 10: Top 10 positive & negative expressions and their score

On the positive side, not much can be concluded, due to the lack on details. Customers that like the trike simply say it is a great product ("*Four Stars*","*Great*,"*Love*"), without saying why. This is totally opposite to the negative side, where much clear expressions appear.

Most Positive	**Most Negative**
Four Stars	Three
Trike	Missing Part
Great	Meh
Tricycle	Dangerous
Love	Frustration
Love the Bike	Beware
Great Bike	Shenanigans

Table 3: Selected relevant positive and negative expressions extracted from the sentiment analysis

The main worries of the users are the missing parts from the trike and the danger or riskiness of the trike. The feelings of the user towards the product are summarized in "Meh" and "Frustration", both probably related to the assembly of the product. It also seems that the trike is dangerous to ride and that users should be aware of possible falls. For this project interest, one of these products weaknesses is starting to show up.

More information and the implementation of this program can be found in the Predicting Sentiment script on the Appendices.

Topic Modeling - LDA

Another way of analyzing the user data is to follow the topic modeling approach. By taking inferred topics and analyzing the sentiment of their corresponding documents (reviews) what customers are saying (or feeling) can be found. Examples from other authors[10,11,12,13].

Latent Dirichlet Allocation (LDA) is the technique used to extract topics from Amazon reviews. It is assumed that there is some number of topics (manually chosen, 10 topics in this case), each topic having associated probability distribution over words and each document has its own probability distribution over topics.

First, it scopes out a bunch of different **documents**, making note of the **words** in each of them. Crucially, it does not know the **topics** of each document, neither the different topics of each word.

Therefore, it picks some number of topics to learn, and starts by making a guess as to why each word belongs to each document. Of course, the random guesses are very likely to be incorrect, but there is a way to improve them, known as Gibbs sampling:

- Go through each word w in document d
- For each topic k, computes two things:
  - p($k \mid d$) = the proportion of words in document d that are currently assigned to topic k
  - p($w \mid k$) = the proportion of assignments to topic k over all documents that come from this word w.
- Reassign word w a new topic, where we choose topic k with probability p($k \mid d$) * p($w \mid k$). According to the generative model, this is essentially the probability that topic k generated word w.

Finally it takes into account the distribution of each topic in each document (with the parameters α and η, leading to:

$$p(\beta, \theta, z, w) = \prod_{i=1}^{K} p(\beta_i) \prod_{d=1}^{D} p(\theta_d) \prod_{n=1}^{N} p(z_{d,n}|\theta_d) p(w_{d,n}|\beta_{1:K}, z_{d,n})$$

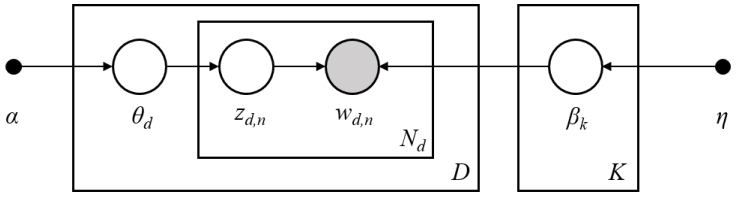


Figure 11: LDA graphical model. Boxes represent replicates. Outer box D are documents, while inner box N_d represents the repeated choice of words within the document

[10] Aashutosh Bhatt, Ankit Patel, Harsh Chheda, and Kiran Gawande. Amazon review classification and sentiment analysis. *International Journal of Computer Science and Information Technologies*, 6(6):5107–5110, 2015. Available at http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.736.4819&rep=rep1&type=pdf

[11] Abhijit Chakankar, Sanjukta Pal Mathur, and Krishna Venuturimilli. Sentiment analysis of usersâĂŹ reviews and comments. Available at https://pdfs.semanticscholar.org/ab8f/62c23b116ab51224e743f4c45871f8bbe995.pdf

[12] Subhabrata Mukherjee and Pushpak Bhattacharyya. *Feature Specific Sentiment Analysis for Product Reviews*, pages 475–487. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012. Available at http://dx.doi.org/10.1007/978-3-642-28604-9_39

[13] Callen Rain. Sentiment analysis in amazon reviews using probabilistic machine learning. *Swarthmore College*, 2013. Available at https://pdfs.semanticscholar.org/f0af/e9ea9d286248336ee9dc4e954aecde3475bb.pdf

Specific Notation:

β_K Topics

θ_d Topic proportions for document d

$z_{d,n}$ Topic assignment for word n in document d

w_d Observed words from document d

η, α, Dirichlet parameters

Summing up, Gibbs sampling[14] basically takes some initial values of parameters and iteratively replaces those values conditioned on its neighbors (Figure 11). Every word in all the text reviews is assigned a topic at random, iterating through each word, weights are constructed for each topic that depend on the current distribution of words and topics in the document, and we iterate through this entire process until it reaches a convergence.

[14] Julian John McAuley. Cse 190 data mining and predictive analytics. lecture 13, sliden63. In *Proceedings of the 22nd international conference on World Wide Web*. UCSD, 2015. Available at http://cseweb.ucsd.edu/~jmcauley/cse190/slides/week8/lecture13.pdf

All the reviews were converted to a bag of words and stop words removed. After the training of the model, a query was made passing a selected string to the model: "issues".

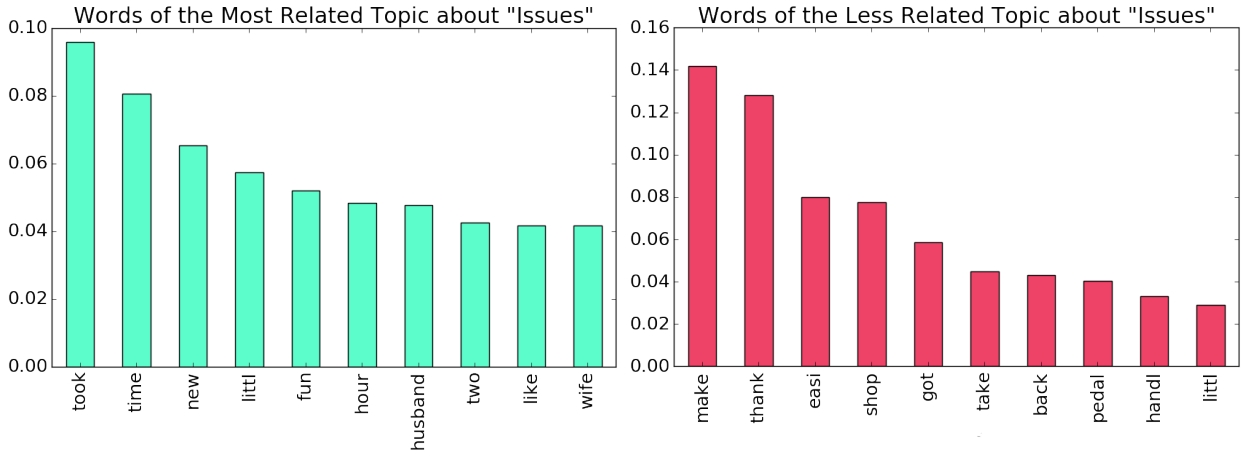


Figure 12: Results from the query to the LDA model: words from the most and less related topics

The most related topic (0.66 correlation) talks about assembly issues, with words like "*took*", "*time*", "*little*", "*fun*", "*husband*", "textit(wife)", whereas the less related topic (0.03 correlation) contains ("*make*", "*thank*", "*easy*", "*shop*", "*bak*"), which has little to do with any issues or problems. In fact, it seems this topic has not a clear meaning behind. Fortunately, both topics do no share words, leading to the conclusion that these topics are far apart, and that are related to different aspects of the products.

As a conclusion to this analysis, the assembly of the bike (that comes in parts) supposes main problem to the users, an issue that will be out of the scope of this project.

Overall, the results are not clear enough to to make an inference about their cause or origin. As the reader can notice, this analysis was not that successful, because the results could be interpreted in various different ways. Having only unigrams does not help to the interpretation of the results. That is why another way was approached, by means of keyword extraction and the wordcloud creation.

More information and the implementation of this program can be found in the Topic Modeling script on the Appendices.

Wordcloud Creation

The Rapid Automatic Keyword Extraction (RAKE)[15] algorithm was implemented to easily extract keywords from the reviews, by identifying runs of non-stopwords and then scoring these phrases across the document. It requires no training, the only input is a list of stop words for a given language, and a tokenizer that splits the text into sentences and sentences into words.

[15] Stuart Rose, Dave Engel, Nick Cramer, and Wendy Cowley. *Automatic Keyword Extraction from Individual Documents*. John Wiley Sons, Ltd, 2010. Available at http://dx.doi.org/10.1002/9780470689646.ch1

A typical keyword extraction algorithm has three main components:

1. **Candidate selection**: extract all possible words, phrases, terms or concepts that can potentially be keywords.
2. **Properties calculation**: For each candidate, calculate properties that indicate that it may be a keyword.
3. **Scoring and selecting keywords**: All candidates can be scored by either combining the properties into a formula, or using a machine learning technique to determine probability of a candidate being a keyword. A score or probability threshold, or a limit on the number of keywords is then used to select the final set of keywords.

Finally, parameters such as the minimum frequency of a candidate, its minimum and maximum length in the words, or the stemmer used to normalize the candidates help tweak the algorithm's performance to a specific dataset.

Figure 13: Unformated wordcloud of extracted keywords

In the Figure 13 there is the first version of the extracted wordcloud, without any format, and Figure 14 is the result after the formatting. The shape of the wordcloud was taken from one of the Amazon products, to simulate the form of a tricycle. More information and the implementation of this program can be found in the Wordcloud Creation script on the Appendices.

From the wordcloud there were positive terms, as "*great bike*", "*front fender*", "*fairly easy*","*highly recommend*". Focusing on the negative features, some interesting terms appeared: "*missing parts*", "*rear fenders*", "*bike shop*","*balance issues*". Out of curiosity, these trikes do not have rear fenders but do have one in the front, leading to the users complains.

As a mechanical engineer, the term "*balance issues*" was the one that attracted my attention. How can a three wheeler bike have balance issues? It is in fact intrinsically more stable than a bike, for sure. However, searching manually through some reviews, I noticed that the origin of this term was in some people complaining about falling when taking a curve, or one of the rear wheels lifting the ground. All these experiences made the users feel insecure when riding the trike. What could be done to avoid this problem?

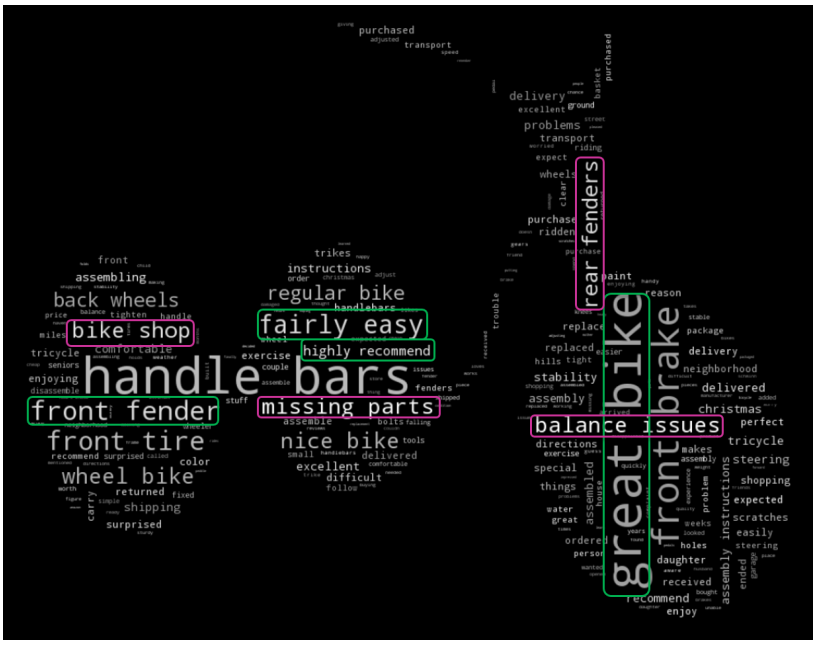


Figure 14: Formated wordcloud of extracted keywords

Narrow Track Vehicles

Balance or stability issues only appear when the user goes into a curve, otherwise the three wheels always support the trike in stable position. Let us analyze these type of vehicles in terms of the stability.

The PEV can be classified as a narrow track vehicle and, as it operates in normal driving conditions, it needs to be relatively tall to assure visibility. In Figure 15 the bounding box dimensions are represented. A tall narrow vehicle is characterized by a high centre of gravity to track width ratio, which makes vehicle **roll stability** an issue[16,17].

When the track width of a vehicle is narrowed, there is an increasing in the the ratio of the center of gravity height with respect to the track width. This particular geometric property poses **stability problems for these types of vehicles while cornering**. Hence, such a narrow vehicle needs to be equipped with an inherent tilting mechanism that enables the vehicle to **tilt into turns while cornering**[18].

The added tilt degree of freedom should not require any stabilizing input from the driver. This brings about the need for an active control system that ensures the tilt stability (appropriate leaning during cornering) without affecting the handling of the vehicle.

Therefore, if such a vehicle is to negotiate curves through out its operating speed range, it needs to have a **capacity to tilt into the curve** like most two-wheeled vehicles. Consequently, an inherent tilting capability is required.

[16] Saeedi, Kazemi, and AWT TAG. Stability of three-wheeled vehicles with and without control system. *International Journal of Automotive Engineering*, 3, 2013. Available at http://www.iust.ac.ir/ijae/article-1-180-en.html

[17] Andrea Festini, Andrea Tonoli, and Enrico Zenerino. *Urban and extra urban vehicles: re-thinking the vehicle design*. INTECH Open Access Publisher, 2011. Available at https://www.intechopen.com/books/howtoreference/new-trends-and-developments-in-automotive-system urban-and-extra-urban-vehicles-re-thinking-the-v

[18] Stephen Nurse, Mark Richardson, and Robbie Napper. Tilting human powered trikes: Principles, designs and new developments. In *Australasian Transport Research Forum (ATRF), 37th, 2015, Sydney, New South Wales, Australia*, 2015. Available at http://atrf.info/papers/2015/index.aspx

Figure 15: PEV bounding box dimensions

This **tilting mechanism** should not require any special skills from the driver. The driver should be able to drive about as he/she would a regular four-wheeled vehicle. This, in turn, brings about a need for an **active tilt control system** that controls the tilting mechanism and ensures the stability of the vehicle and safety of the driver.

Stability

Narrow vehicles that do not lean are unstable when they corner too hard. A simple statics calculation can illustrate this fact (Figure 16).

As the vehicle accelerates, for example, in a turn to the right, due to increasing lateral force F_y, the reaction in the right wheel R_r approaches zero while the reaction in the left one R_l approaches the weight force mg.

At the balance limit,

$$T_r = \frac{bmg}{2}$$

and

$$T_f = 2hF_y$$

Overturning occurs if $T_r < T_f$, or similarly, the wheelbase is limited to

$$b < \frac{4hF_y}{mg}$$

By introducing tilt control, the **center of gravity is shifted** so that R_r never reduces to zero.

h_g Height of the center of gravity

b Wheelbase width

R_l, R_r Left and right wheel reactions

F_y Lateral force

T_r Moment due to the reactions R_l, R_r

T_f Moment due to lateral force F_y

m Mass of the vehicle

g Earth gravity acceleration

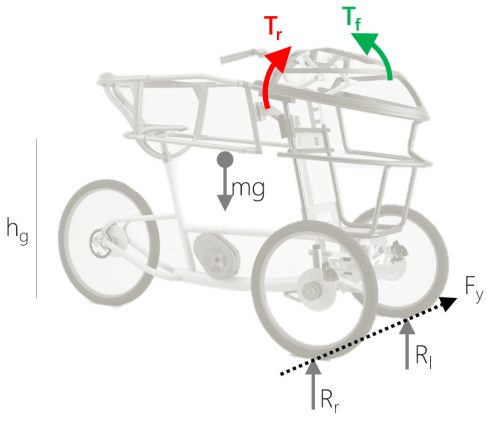


Figure 16: Force schematic in a right turn scenario

Research Objectives

This project is going to consist on the **design and fabrication of a three wheeler vehicle with an active tilting system**. The aim of the work presented here is summarized as follows:

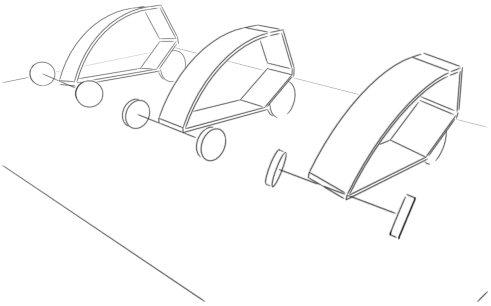


Figure 17: Simple first sketches of a tilting PEV

- The main objective was to *design and fabricate a three-wheeled tilting vehicle* and to *implement a roll control system*. In this way, due to the size, weight and cost limitations of a lightweight vehicle like PEV, it was critical to constrain the size of the motors and the feasible mechanical components.

- The first aim was to *review the current developments* in vehicle modeling, roll control and tire modeling. Understanding the previous work in this field is fundamental to know the limitations and problems of the applied techniques.

- The second aim was to implement and investigate models and methods to quantify uncertainties in the vehicle system such as the kinematics and environmental factors. These factors had a direct effect on the experimental results of road tests as well as the vehicle modeling and simulation.

- The final aim was to set up an experimental test program to further the understanding of the dynamic model and the control of the three-wheeled tilting vehicle and to obtain data for the validation of the vehicle model.

Thesis Structure

This thesis consists of eight chapters and the following section outlines the structure of the chapters.

- **Chapter 2** reviews the literature relevant to all the aspects of this thesis: *vehicle and roll dynamics modelling, roll control methods, driver modelling and tyre modelling*. The literature is critically discussed and a number of roll control methods, bicycle control strategies and rider models have been replicated. This chapter illustrates the status quo in the various research fields, highlights the gaps in the current knowledge.

- In **chapter 3** a *linear vehicle model* is presented along with the proposed control method. It covers from the dynamic vehicle model to the calculations and simulations to test theoretically the control system. In order to improve the dynamic performance of the vehicle, *linear optimal control theory* was used, and some design parameters for the control algorithm are presented.

- **Chapter 4** explains the development of an small scale prototype of a tilting three wheeler vehicle. The components and design process are also documented in this section.

- **Chapter 5** presents the developed *simulations to design the front suspension* and describes the *fabrication process to build a real scale* tilting trike. The selected components and the electronic schematic is also exhaustively explained.

- **Chapter 6** describes the application of the control strategy to the designed vehicle, as well as the *experimental set up and the results from the carried out tests*.

- **Chapter 7** summarizes the *materials, machining and labor cost* to carry out the project.

- Finally, **chapter 8** *concludes the thesis* and presents a number of thoughts on future work.

2. Background

The aim of this chapter is to critically review the literature relevant to the study of tilting three wheelers and their control strategies.

Active and Passive Tilting

The dynamics of a tilting PEV are quite different from a conventional car. Indeed, due to the narrow track of these type of vehicles, and according to the position of their **center of gravity**, their dynamics on turning may be closer to the dynamics of two-wheeler vehicles such as motorcycles.

Thus, some of these vehicles have to tilt when turning to compensate for the effect of lateral acceleration and remain in balance. In the case of a motorcycle, it is the driver who causes the lateral movement of the motorcycle (**passive tilting**), whereas for heavy and bulky vehicles, the inclination must be automatic (**active tilting**).

The main drawback with a passive tilting system is the necessity of a **locking mechanism to avoid the leaning** of the vehicle when standing. On the contrary, an active system does not need a locking mechanism, since is an actuated mechanism. In addition, the ability to control the leaning can facilitate users to get on the vehicle by leaning to one side and returning to the vertical position to start the ride.

Theferore, this problem of **stability in the curves** is a major technological challenge for these narrow vehicles. Some manufacturers have solved the difficulty by lowering the center of gravity of the vehicle to a minimum (see Tango[19] or Twizy[20]), for example by placing the batteries very low if it is an electric vehicle.

In the case of narrow active tilting vehicles, different actuating and control strategies have been studied in the industry, which are presented below (DTC, STC or SDTC).

Figure 18: Passive tilting system from *Tilting Motor Works*

Figure 19: Active tilting system from the *Toyota i-ROAD concept car*

[19] Commuter Cars. Tango - lowest center of gravity of any car. http://www.commutercars.com/

[20] Renault. Twizy - compact electric vehicle. https://www.renault.co.uk/vehicles/new-vehicles/twizy.html

Narrow Three-Wheeled Vehicles

In this section narrow tilting vehicles with three wheels are presented. In order to easily identify the different types, the following notation is adopted: **[nF, tT, pP, S]** with n the number of front wheels, t the number of tilting wheels, p the number of passengers and S the maximum speed in Km/h.

- **GM Lean Machine**[21] (1F, 1T, 1P, 125) was developed by Frank Winchell of General Motors in the early 1980's. The lateral stability of the vehicle had to be ensured by the driver, through a pedal that controlled the tilt.

- **Mercedes Life Jet F300**[22] (2F, 3T, 2P, 210) was introduced in 1997 by Mercedes-Benz at the Frankfurt Motor Show. The tilt was automated through hydraulic actuators.

- **CLEVER**[23] (1F, 1T, 2P, 80), was developed at the University of Bath (UK) in collaboration with BMW. The inclination of the front module of the vehicle is possible thanks to the hydraulic actuators linking the cab to the non-tilting rear part. The first prototypes were built in 2006, but the transient behavior of this vehicle did not always guarantee lateral stability.

- **Carver One**[24] (1F, 1T, 2P, 185), a three-wheel tilting vehicle, born from a first aerodynamic concept. The Carver One is equipped with an automotive power unit and therefore with a wide non-tilting rear axle. The dynamics of the vehicle are based on the patented 'Dynamic Vehicle Control' technology, which is said to ensure vehicle stability by tilting the vehicle.

- **Piaggio MP3 Hybrid 300IE**[25] (2F, 3T, 1P, 145), is a three-wheeled scooter with a parallel "hybrid" propulsion, based on a 125 cc engine and a 3.4 hp electric motor. Like a traditional scooter, this vehicle is tilted directly by the driver.

Paying attention to more lightweight vehicles (bike-like):

- **Veleon**[26] (2F, 3T, 1P, 25) both cargo and driver lean into the curve, helping to whoosh cars and other obstacles, including tight corners at high speed. The mode can be deactivated by pressing a button and the bicycle stays in vertical position.

- **Trego**[27] (2F, 3T, 1P, 25) is a successful fusion of a cargo bike and a folding bike, a modular separable tricycle. Combines a rear wheel to two front wheels, that are connected by an L structure, being extremely useful for transporting bulky items.

[21] General Motors. Lean machine. http://www.carstyling.ru/en/car/1982_gm_lean_machine/

[22] Mercedes Benz. Life jet f300. http://www.evo.co.uk/mercedes/18241/concept-cars-the-mercedes-benz-f300-life-jet

[23] University of Bath and BMW. Clever. http://www.bmwblog.com/2009/10/09/bmw-unveils-clever-concept/

[24] Vehicle Class, C. R. Van Den Brink, and H. M. Kroonen. *AVEC '04 DVC - The banking technology driving the CARVER*, 2004. Available at http://www.carver-technology.com/PDF/brink_AVEC_2004.pdf

[25] Piaggio. Mp3 hybrid 300ie. http://www.piaggio.com/mp3/en_EN/models/hybrid/hybrid-300IE/

[26] Adomeit Group GmbH. Veleon. http://www.veleon.de/eng/veleon.htm#vielfalt

[27] Trego. http://trego-trolley.com/

– **Kaylad-e**[28] (2F, 3T, 1P, 25) is a tilting trike concept with a 250-W electric motor and a lithium battery. The concept also includes a secure luggage holder or and optional heavy-duty container that can be mounted on the front of the bicycle.

– **Carqon**[29] (2F, 3T, 1P, 30) has a patented carving mechanism underneath the frame and is equipped with a high torque motor for maximum torque and performance. It is similar to riding a normal bicycle where the driver uses his/her body to control the steering more than actually the turning of the handlebar.

[28] Dimitris-Niavis. Kaylad-e. http://newatlas.com/kaylad-2-electric-tricycle/23049/

[29] Protanium. Carqon. http://www.carqon.com/

Figure 20: Tilting Vehicles

a) GM Lean Machine

b) Mercedes Life Jet F300

c) CLEVER

d) Carver One

e) Piaggio MP3 Hybrid 300IE

f) Veleon

g) Trego

h) Kaylad-e

i) Carqon

Tilt-Actuating Architectures

Direct Tilt Control (DTC)

The inclination is achieved by means of actuators mounted directly on the longitudinal axis of the vehicle, **providing a torque** (M_t) to tilt the whole vehicle or some parts of it. It is the most used system on narrow tilting vehicles and the simplest active system to implement.

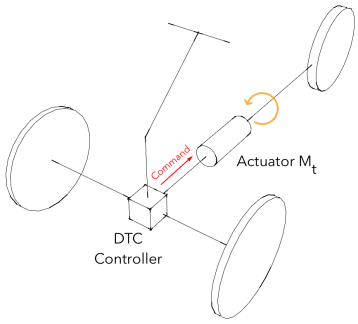


Figure 21: DTC system schematic

In a turn, the lateral acceleration is proportional to the longitudinal velocity, as is demonstrated in the next chapter. Therefore, for the same steering angle, the lateral acceleration is higher at high speed. As a result, the **torque** M_t **required is greater at high speed** since it must ensure the desired inclination in the opposite direction to that of the lateral acceleration.

It is at high speed that there is a greater risk of **saturation of the actuator**. It should be noted that if the DTC system only acts from the measurement of the measured acceleration, this may cause an unpleasant feeling in the passengers. When driving a motorcycle, the rider tilts the vehicle simultaneously with the taking of the turn, so that the perceived acceleration remains virtually zero, even in transient situations.

Steering Tilt Control (STC)

These systems control the steering angle of the wheels. The steering angle (δ_{driv}) applied by the driver is modulated by the STC system (δ_c) to control the tilt angle using countersteering. This strategy is inspired by the action of a bicycle or motorcycle rider, but is seldom used by manufacturers, since it is based on a **steer-by-wire** system, which is still prohibited to commercialize for safety standards reasons.

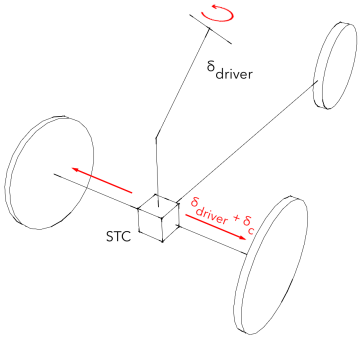


Figure 22: STC system schematic

STC systems are not well suited for **low longitudinal speeds**, demanding a large counter-steering to tilt the vehicle, which deviates it significantly from its trajectory. In contrast, the STC system is more efficient than the DTC at high speed, as a large torque is required by the DTC (due to the delay in the DTC tilting torque).

Comparison between DTC and STC

Let us compare both tilting systems by looking to the forces actuating in each case. The vehicle is traveling in an straight line at a constant velocity V, when a right turn comes.

- DTC: the driver will turn the steering wheel to the right (Figure 23), producing a lateral acceleration in the inner direction of the curve(do not confuse it with the inertial force which pushes the vehicle outwards the curve). Due to the possibility of tilting, the vehicle will tend to lean to the left, but in this moment the DTC torque M_t will correctly lean it to the right. In this simple case it is clear the delay between the lateral acceleration and the torque M_t.

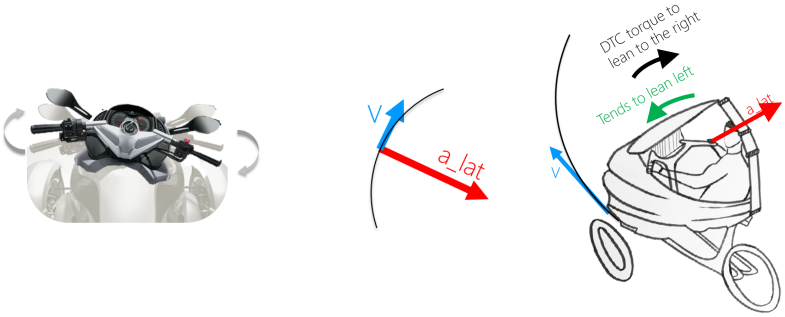


Figure 23: Direct Tilt Control
a) Steering input from the driver
b) DTC force diagram in a right turn
c) DTC tilting torque input

- STC: the STC steer-by-wire system will slightly turn the steering wheel to the left (Figure 24), producing a lateral acceleration in the outer direction of the curve. Due to the possibility of tilting, the vehicle will correctly lean to the right. By using the countersteering, the vehicle easyly leans to the appropiate side, but the desired path of the user is not perfectly followed.

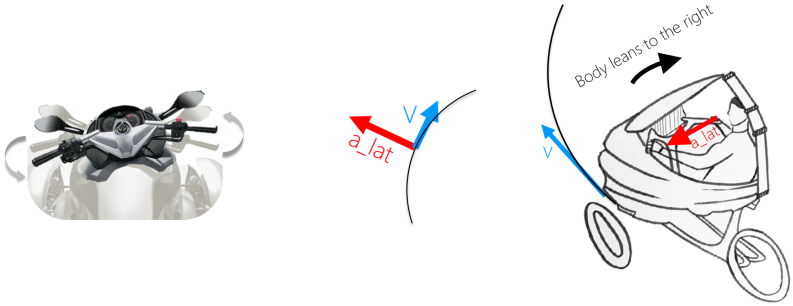


Figure 24: Steering Tilt Control:
a) Steering input from the actuator
b) STC force diagram in a right turn
c) STC tilts automatically in a right turn

In order to give an overall view, the table 4 summarizes the characteristics of the two systems DTC and STC.

	DTC	**STC**
Lean by	Roll actuator	Countersteering
Stability	Always stable	Unstable at low speeds
High Speed	Deal with high lateral acceleration	Small adjustments
Low Speed	Good behaviour at low speeds: $M_t \Uparrow$	Large and frecuent steer inputs $\delta_c \Uparrow$

Table 4: DTC vs STC comparison table

SDTC hybrid solution

To take advantage of both strategies and their complementary efficiencies according to speed, recent work has focused on the so-called SDTC hybrid solution for vehicles with both a tilt actuator and a steer-by-wire system. The first solutions are monovariable, meaning that a single control input is active at a time, with switching from one system to the other as a function of the speed.

The STC systems are generally implemented from a threshold speed of approximately $V = 40Km/h$ while the DTC systems are active below that speed. Switching from one system to another by switching is complicated due to the fact that the control signals act as antagonist in transient phases[30],[31].

- In transient, the STC system causes a counter-steering, and the created lateral acceleration created serves for the inclination of the vehicle. Switching from STC to DTC, the DTC command will seek to tilt the vehicle in the opposite direction to the perceived lateral acceleration, hence in the wrong direction. The two control signals are conflicting or antagonistic in this case.

- Switching from DTC to STC, the control signals are not conflicting, but the tilt torque DTC M_t abruptly cancels out as the vehicle starts to tilt. This interruption creates a discontinuity and peaks in the signals.

[30] Sang-Gyun So and Dean Karnoop. Active dual mode tilt control for narrow ground vehicles. *Vehicle System Dynamics*, 27(1):19–36, 1997. Available at http://dx.doi.org/10.1080/00423119708969321

[31] Robin Hibbard and Dean Karnopp. Twenty first century transportation system solutions-a new type of small, relatively tall and narrow active tilting commuter vehicle. *Vehicle system dynamics*, 25(5):321–347, 1996. Available at http://dx.doi.org/10.1080/00423119608968970

Vehicle Model

Modeling is a key step in the study and control of systems. Depending on the use to be made (embedded model, design model, simulation model...), different levels of complexity can be envisaged.

Models Used in the Literature

The study and development of narrow vehicles only began recently, due to the congestion problems of road traffic and the increase in pollution. The problem of narrow and tilting vehicles is therefore recent, and few research teams have worked on this subject. The **narrow tilting vehicle models** will be reviewed in this section, explaining their advantages and limitations.

Researchers at **California University** were among the first to take an interest in this matter. In Hibbard R. *et al.*, (1996)[32] and So S. *et al.*, (1997a)[33], the authors consider only the dynamics of the angle of inclination θ, and the model used for the simulation was reduced to the transfer function between the input M_t and θ.

A more detailed linearized simulation model at small angles, with longitudinal dynamics, was used in later work (So S. *et al.*, 1997b)[34].

In the UK, at the **University of Bath**, the researchers worked on the CLEVER project. In his thesis[35], J. Berote focused on the vehicle modeling, developing a 5 degree of freedom (dof) multi-body model, that included dynamics of hydraulic actuators, which was then incorporated into the SimMechanics software to obtain a validation model.

Previously, they used a PID controller (J. Berote *et al.*, 2008)[36], that did not require any model. For such a purpose, the authors considered approximate relations between the steering angle and the lateral acceleration, and between the steering angle and the angle of inclination.

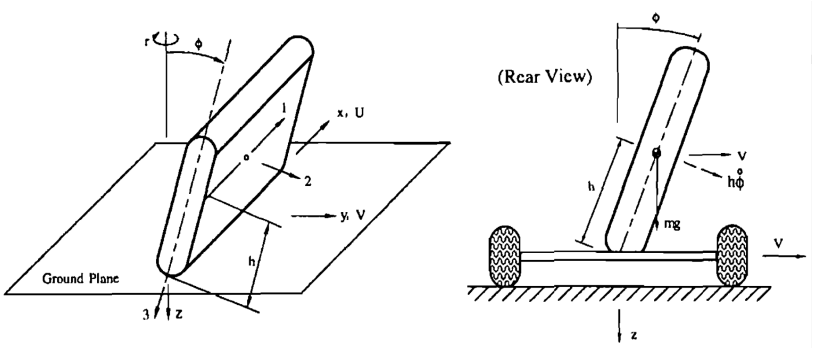


Figure 25: Literature model: Inverted pendulum simple model for tilting vehicles

[32] Robin Hibbard and Dean Karnopp. Twenty first century transportation system solutions-a new type of small, relatively tall and narrow active tilting commuter vehicle. *Vehicle system dynamics*, 25(5):321–347, 1996. Available at http://dx.doi.org/10.1080/00423119608968970

[33] Sang-Gyun So and Dean Karnoop. Active dual mode tilt control for narrow ground vehicles. *Vehicle System Dynamics*, 27(1):19–36, 1997. Available at http://dx.doi.org/10.1080/00423119708969321

[34] Sang-Gyun So and Dean Karnopp. Switching strategies for narrow ground vehicles with dual mode automatic tilt control. *International Journal of Vehicle Design*, 18(5):518–532, 1997. Url:http://www.inderscienceonline.com/doi/abs/10.1504/IJVD.1997.062071

[35] Johan J H Berote. Dynamics and control of a tilting three wheeled vehicle. Available at http://opus.bath.ac.uk/24680/, 2010

[36] Johan Berote, Auguste Van Poelgeest, Jocelyn Darling, Kevin A Edge, and Andrew Plummer. The dynamics of a three-wheeled narrow-track tilting vehicle. In *FISITA World Automotive Congress 2008*, September 2008. Paper number: F2008-SC-032. Available at http://opus.bath.ac.uk/15090/

At the **University of Minneapolis** in Minnesota, a prototype recliner was built, and several advances were made. In the article R. Rajamani *et al.*, (2003)[37] the authors focus on modeling the vehicle. They neglect the longitudinal dynamics, but take into account the gyroscopic moments of the wheels and the tilting moment.

Their synthesis model for the design of the control law, also used for simulation, was later simplified and a linearized model of equation (Piyabongkran D. *et al.* 2004[38], Kidane S. *et al.*, 2010[39], Kidane S. *et al.*, 2008[40]).

[37] R. Rajamani, J. Gohl, L. Alexander, and P. Starr. Dynamics of narrow tilting vehicles. *Mathematical and Computer Modelling of Dynamical Systems*, 9(2):209–231, 2003. Available at http://www.tandfonline.com/doi/abs/10.1076/mcmd.9.2.209.16521

[38] D Piyabongkarn, T Keviczky, and R Rajamani. Active direct tilt control for stability enhancement of a narrow commuter vehicle. *International Journal of Automotive Technology*, 5(2):77–88, 2004. Available at https://pdfs.semanticscholar.org/b079/59e2ab35f552f1b06209af731e148e32a0ec.pdf

[39] S. Kidane, R. Rajamani, L. Alexander, P. J. Starr, and M. Donath. Development and experimental evaluation of a tilt stability control system for narrow commuter vehicles. *IEEE Transactions on Control Systems Technology*, 18(6):1266–1279, Nov 2010. Available at http://ieeexplore.ieee.org/document/5356230/

[40] S. Kidane, L. Alexander, R. Rajamani, P. Starr, and M. Donath. A fundamental investigation of tilt control systems for narrow commuter vehicles. *Vehicle System Dynamics*, 46(4):295–322, 2008. Available at http://www.tandfonline.com/doi/abs/10.1080/00423110701352987

Bicycle Model

The lateral control of the tilting vehicles is the object of this study. The so-called **bicycle model** has been chosen to be considered as a conceptual model which aggregates the 2 front wheels in one wheel, and only one rear wheel. Wheel torques are also simplified. They are implicitly replaced by the steering angle.

This model has been built from the 4 dof model presented in the article of R. Rajamani *et al.*, (2003)[19], which considers the longitudinal speed as a parameter of the model and not as a state variable.

The model obtained takes into account the longitudinal dynamics relating to the speed, as well as the lateral dynamics including the inclination. Figures 26 and 27 respectively show the views in the horizontal (XY) and vertical (YZ) planes of the vehicle.

The quantities and notations used below are defined in the 'Notations' section at the beginning of the thesis. Let us define the following coordinate systems:

1. The absolute coordinate system R=(X, Y, Z) fixed to the ground, the axes X and Y are on the ground while the Z is the vertical axis.

2. The vehicle coordinate system r=(x, y, z) is linked to the vehicle. Its origin is the point G (center of gravity), the axis x is parallel to the longitudinal axis of the vehicle, the axis z is parallel to the axis Z and the axis y is such that the planes xy and XY are parallel.

3. The coordinate system r'=(x', y', z') is linked to the frame and tilts with it. It coincides with the reference mark (x, y, z) when the vehicle is in vertical position. Its axes y' and z' are inclined by an angle θ equal to the angle of inclination of the vehicle.

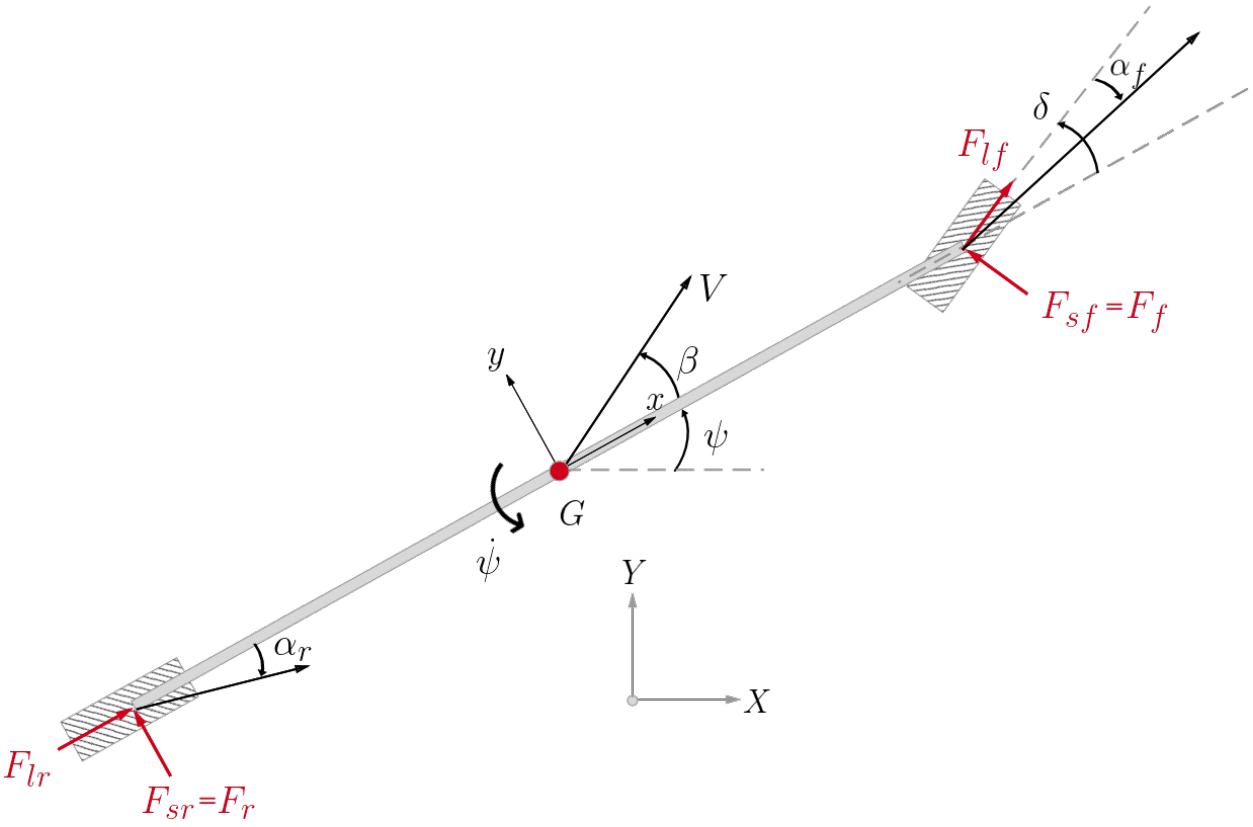


Figure 26: Bicycle model: View of the plane XY

The vehicle has 3 degrees of freedom (listed below) and takes into account the effects of trail, slip angles and gyroscopic effects.

y Lateral position with respect to road reference
ψ Yaw angle
θ Tilt angle

Model Equations

The equations of the model implemented in this thesis have been derived using one principle:

– The **fundamental principle of dynamics in translation and rotation**, also called Newton's laws, which require knowledge of the mechanical connections of all internal and external forces acting on the vehicle, and their points of application:

$$\sum_i F = ma, \quad \sum_j M = I\ddot{\theta}$$

Where F represents the external forces applied to the body, i the number of these forces, m its mass and a its acceleration. M the external moments applied to the body, j the number of these moments, I its inertia, and $\ddot{\theta}$ its angular acceleration.

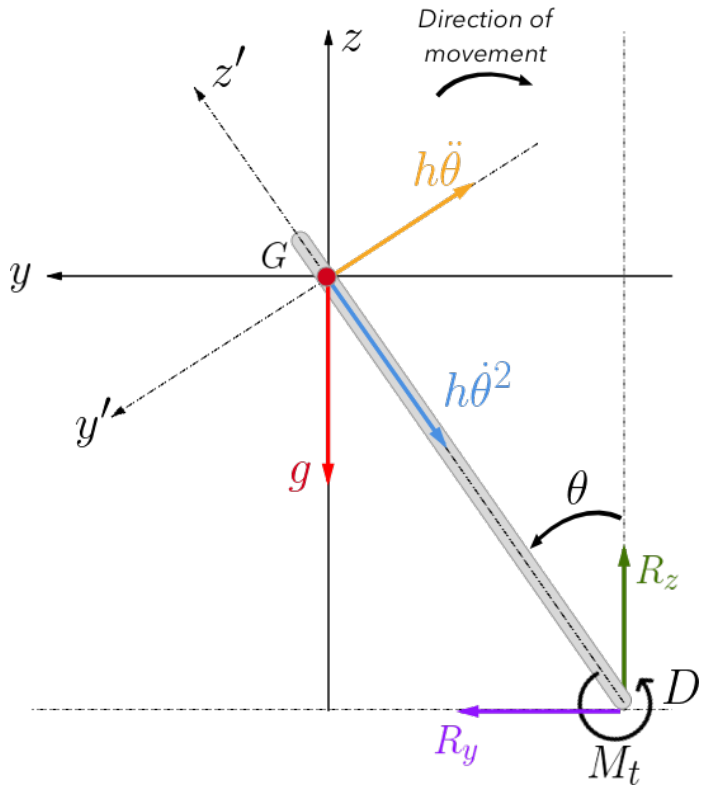


Figure 27: Bicycle model: View of the plane YZ

Applying Newton's principle to the presented bicycle model:

$$\sum_i \vec{F} = m\vec{a} \qquad y) \quad R_y = m\,a_y \qquad (1)$$

$$z) \quad R_z - mg = m\,a_z \qquad (2)$$

$$\sum_j \vec{M} = I\vec{\ddot{\theta}} \qquad x) \quad I_x\ddot{\theta} = R_z\,h\,\sin\theta \;-\; R_y\,h\,\cos\theta \qquad (3)$$

$$z) \quad I_z\ddot{\psi} = F_f\,L_f - F_r\,L_r \qquad (4)$$

The absolute acceleration (in the inertial reference frame R) of a moving body can be decoupled in two terms if a non-inertial frame of reference r is attached to the body.

Hence, the accelerations a_y and a_z can be obtained from the acceleration of the point D (in contact with the ground), plus the relative accelerations of the center of gravity G with respect to the point D:

$$a_y = a_{D,y} + a_{G/D,y} \qquad (5)$$

$$a_z = a_{D,z} + a_{G/D,z} \qquad (6)$$

The center of gravity G describes a circular arc of center D at speed $v_G = h\dot{\theta}$ with $\bar{DG} = h$. The accelerations experienced at G are the tangential acceleration $\frac{dv_G}{dt} = h\ddot{\theta}$ along the axis y', and the normal acceleration $-h\dot{\theta}^2$ along the axis z'. By projecting these equations on the axes of the reference r=(x, y, z):

$$a_{D,y} = \ddot{y} + V\dot{\psi}$$

$$a_{D,z} = 0$$

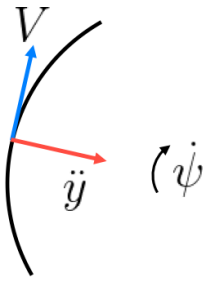


Figure 28: Circular path velocity and lateral acceleration

If D is taken into account as a fixed point, the relative motion can be considered:

$$v_G = v_D + \vec{w} \wedge \vec{DG} = 0 + \dot{\theta}\, h(\cos\theta\, \vec{j} - \sin\theta\, \vec{k})$$

$$a_G = a_D + \vec{\alpha} \wedge \vec{DG} - w^2\, \vec{DG} = 0 + \ddot{\theta} h(\cos\theta\, \vec{j} - \sin\theta\, \vec{k}) - \theta^2 h(\sin\theta\, \vec{j} - \cos\theta\, \vec{k})$$

$$a_{G/D,y} = \ddot{\theta} h \cos\theta - \theta^2 h \sin\theta$$

$$a_{G/D,z} = -\ddot{\theta} h \sin\theta - \theta^2 h \cos\theta$$

Having studied the relative motion, the absolute accelerations are expressed as:

$$a_y = a_{D,y} + a_{G/D,y} = \ddot{y} + V\dot{\psi} + \ddot{\theta} h \cos\theta - \theta^2 h \sin\theta \tag{7}$$

$$a_z = a_{D,z} + a_{G/D,z} = -\ddot{\theta} h \sin\theta - \theta^2 h \cos\theta \tag{8}$$

Therefore, replacing these expressions into equations (1), (2) and (3):

$$R_y = m\, a_y = m(\ddot{y} + V\dot{\psi} + \ddot{\theta} h \cos\theta - \theta^2 h \sin\theta)$$

$$R_z = m\,(a_z + g) = m(-\ddot{\theta} h \sin\theta - \theta^2 h \cos\theta + g)$$

$$I_x\ddot{\theta} = mh(-\ddot{\theta} h \sin^2\theta - \theta^2 h \sin\theta\cos\theta + g\sin\theta - \ddot{y}\cos\theta - V\dot{\psi}\cos\theta - \ddot{\theta} h \cos^2\theta + \theta^2 h \sin\theta\cos\theta)$$

The last equation represents the roll motion, and since it is going to be actuated by the motor, the tilting torque M_t has been included.

$$I_x\ddot{\theta} = mh(-\ddot{\theta} h + g\sin\theta - \ddot{y}\cos\theta - V\dot{\psi}\cos\theta) + M_t$$

To get equation (4), the moment equilibrium is calculated in the c.g. Figure 29 illustrates the vehicle from the top view, with the lateral tire forces F_f and F_r.

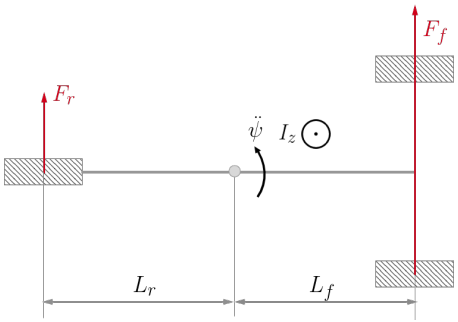


Figure 29: Lateral tire forces and yaw dynamics

$$I_z\ddot{\psi} = F_f\, L_f - F_r\, L_r \tag{9}$$

The lateral force exerted on a wheel is expressed as $F = C\alpha + \lambda\theta$, where C is the coefficient of stiffness and $\lambda\theta$ is due to the camber. Using the equality $\alpha = \delta - \tan^{-1}\left(V_y/V_x\right)$, at the level of the front wheels, the lateral tire forces are defined as follow for small slip angles:

$$F_f = 2C_f\left(\delta - \frac{\dot{y} + L_f\dot{\psi}}{V}\right) + 2\lambda_f\theta \tag{10}$$

$$F_r = C_r\left(-\frac{\dot{y} - L_r\dot{\psi}}{V}\right) + \lambda_r\theta \tag{11}$$

where C_f and C_r are the cornering stiffness of the front and rear tires, and λ_f and λ_r the camber stiffness of each wheel.

Front Wheel Dynamics due to Trail

The axis of the wheel has so far been considered perpendicular to the plane of the ground. This is not really the case in practice, since the axis is generally slightly inclined as shown in Figure 30.

In absence of suspension dynamics, tilt dynamics and longitudinal acceleration, the sum of the normal forces will equal the weight of the vehicle. In addition, considering the similarity relation from Figure 31, the normal force acting on the front wheels F_z can be obtained:

$$\frac{F_{rz} + F_{fz}}{L_r + L_f} = \frac{F_{rz}}{L_r} = \frac{F_{fz}}{L_f}$$

$$F_{fz} + F_{rz} = mg \quad \Rightarrow \quad F_{fz} = F_z = mg\frac{L_r}{L_r + L_f} \tag{12}$$

Thus, the reaction of the ground F_z, perpendicular to the plane of the ground, will contribute with a moment equal to $F_z \sin\theta$ to the steering axis. Similarly, the lateral force also contributes a restoring steering torque on the wheel (Figure 32):

$$M_{trail} = F_z \gamma \cos\beta \sin\theta - F_f \cos\theta(\gamma + \gamma_{pneumatic}) \cos\beta \tag{13}$$

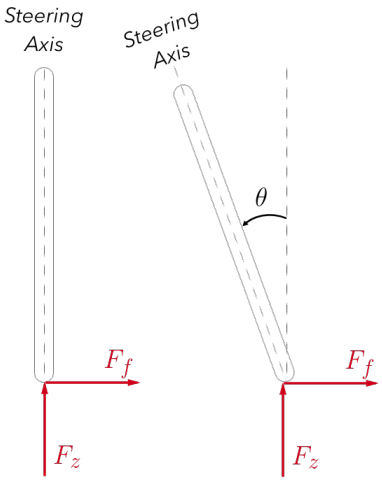


Figure 30: Inclined steering axis due to the tilting

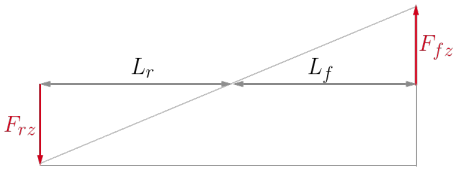


Figure 31: Vertical forces diagram

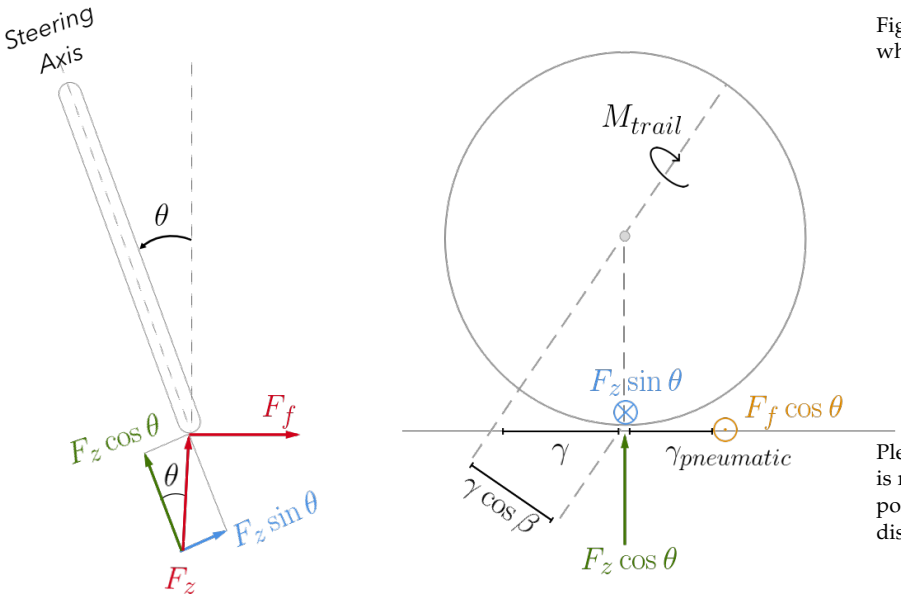


Figure 32: Forces on front wheels when the vehicle is tilted to the left

Please notice, that the lateral force F_f is not located exactly in the contact point with the ground, there exists a distance $\gamma_{pneumatic}$.

Gyroscopic Moments

The dynamic equations of a rigid body with 3 rotational degrees of freedom are derived from the Euler equations:

$$x) \quad \sum \vec{M}_x = I_{xx}\,\dot{\vec{w}}_x - (I_{yy} - I_{zz})\vec{w}_y\vec{w}_z \tag{14}$$

$$y) \quad \sum \vec{M}_y = I_{yy}\,\dot{\vec{w}}_y - (I_{zz} - I_{xx})\vec{w}_x\vec{w}_z \tag{15}$$

$$z) \quad \sum \vec{M}_z = I_{zz}\,\dot{\vec{w}}_z - (I_{xx} - I_{yy})\vec{w}_x\vec{w}_y \tag{16}$$

Inertial reference frame:

$$\sum \vec{M} = \frac{d\vec{H}}{dt} = \frac{d(I\vec{w})}{dt}$$

Non-inertial reference frame:

$$\sum \vec{M} = \left(\frac{d(I\vec{w})}{dt}\right)_{fixed} + \vec{w} \wedge (I\vec{w})$$

The vehicle has two rotational degrees of freedom: yaw and roll (pitch is neglected). In addition, each rotating wheel has three rotational degrees of freedom, and Euler's equations should be applied to the rotating wheels.

Free body diagrams of the front wheels, rear wheel and vehicle body are shown in Figure 35.

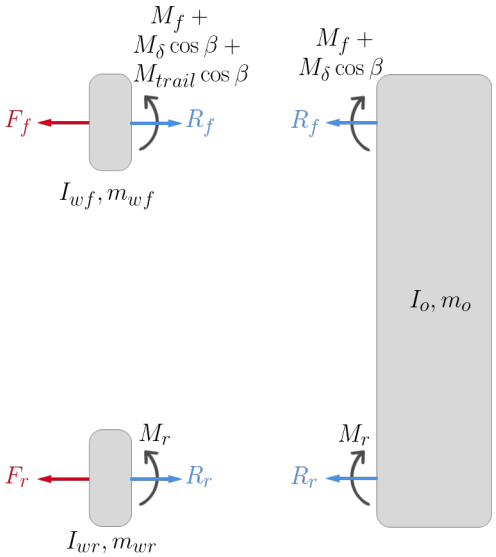


Figure 33: Free body diagram of front wheels, rear wheels and body

- Front Wheels

$$F_y) \quad 2\,m_{wf}\left(\ddot{y} + V\dot{\psi} + h_f\ddot{\theta}\cos\theta - h_f\dot{\theta}^2\sin\theta + L_f\ddot{\psi}\right) = F_f - R_f \quad (17)$$

$$M_z) \quad 2\,I_{wf,\psi}(\ddot{\psi} + \ddot{\delta}\cos\beta) + 2(I_{wf,\theta} - I_{wf,rot})w_{rot}(\dot{\theta} - \dot{\delta}\sin\beta) = M_f + M_\delta\cos\beta + M_{trail}\cos\beta \quad (18)$$

- Rear Wheels

$$F_y) \quad m_{wr}\left(\ddot{y} + V\dot{\psi} + h_r\ddot{\theta}\cos\theta - h_r\dot{\theta}^2\sin\theta - L_r\ddot{\psi}\right) = F_r - R_r \quad (19)$$

$$M_z) \quad I_{wr,\psi}\ddot{\psi} + (I_{wr,\theta} - I_{wr,rot})w_{rot}\dot{\theta} = M_r \quad (20)$$

- Body

$$F_y) \quad m_o\left(\ddot{y} + V\dot{\psi} + h_o\ddot{\theta}\cos\theta - h_o\dot{\theta}^2\sin\theta + l\ddot{\psi}\right) = R_f + R_r$$

Replacing R_f and R_r from front and rear wheels equations (17) and (19) onto the latter:

$$m_o\left(\ddot{y} + V\dot{\psi} + h_o\ddot{\theta}\cos\theta - h_o\dot{\theta}^2\sin\theta + l\ddot{\psi}\right) = F_f - 2\,m_{wf}\left(\ddot{y} + V\dot{\psi} + h_f\ddot{\theta}\cos\theta - h_f\dot{\theta}^2\sin\theta + L_f\ddot{\psi}\right) + F_r - m_{wr}\left(\ddot{y} + V\dot{\psi} + h_r\ddot{\theta}\cos\theta - h_r\dot{\theta}^2\sin\theta - L_r\ddot{\psi}\right)$$

From the center of gravity property, some relations can be extracted:

- Longitudinal center of gravity:

$$2\,m_{wf}\,L_f + m_o\,l - m_{wr}\,L_r = 0 \quad \Rightarrow \quad l = \frac{m_{wr}\,L_r - 2\,m_{wf}\,L_f}{m_o}$$

- Vertical center of gravity:

$$m\,h = m_o\,ho + 2\,m_{wf}\,h_f + m_{wr}\,h_r$$

Using the relations derived above (from c.g) and simplifying:

$$m(\ddot{y} + V\dot{\psi} + \ddot{\theta}h\cos\theta - \theta^2 h\sin\theta) = F_f + F_r \quad (21)$$

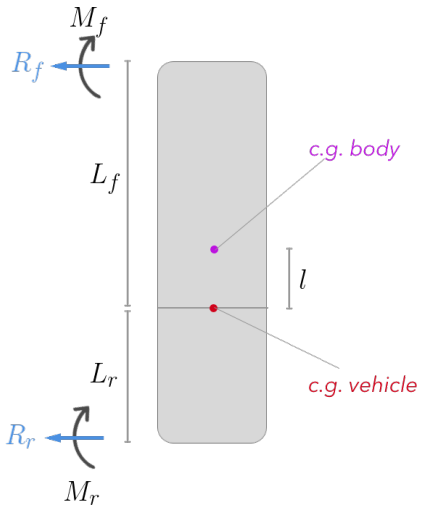


Figure 34: Free body diagram of the vehicle body

$$\begin{aligned} M_z) \quad I_o\,\ddot{\psi} &= -M_f - M\delta\cos\beta - M_r + (L_f - l)R_f - (L_r + l)R_r \\ &= -M_\delta\cos\beta - 2\,I_{wf,\psi}\ddot{\psi} - I_{wr,\psi}\ddot{\psi} - (I_{wr,\theta} - I_{wr,rot}w_{rot}\dot{\theta} + \\ &\quad + (L_f - l)\Big(F_f - 2\,m_{wf}\left(\ddot{y} + V\dot{\psi} + h_f\ddot{\theta}\cos\theta - h_f\dot{\theta}^2\sin\theta + L_f\ddot{\psi}\right)\Big) - \\ &\quad - (L_r + l)\Big(F_r - m_{wr}\left(\ddot{y} + V\dot{\psi} + h_r\ddot{\theta}\cos\theta - h_r\dot{\theta}^2\sin\theta - L_r\ddot{\psi}\right)\Big) \end{aligned}$$

By means of the relations derived above and simplifying, the yaw dynamics equation is obtained:

$$I_z\,\ddot{\psi} = -(I_{wr,\theta} - I_{wr,rot})w_{rot}\dot{\theta} - M_\delta \cos\delta + L_f F_f - L_r L_r \tag{22}$$

Thus the yaw dynamics have changed due to the gyroscopic terms from the rear wheel and the steering torque from the front wheels. The gyroscopic terms due to the front wheels do not appear directly in the vehicle yaw dynamic equations. Following the same procedure, the contribution to the tilt equations can be similarly derived.

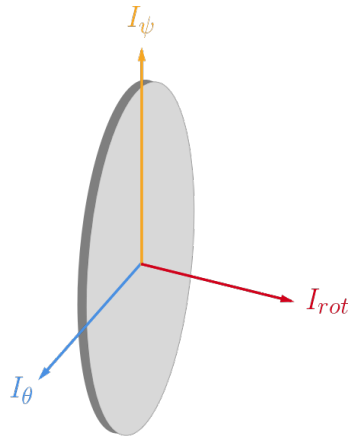


Figure 35: Front wheels axes and inertia notation

- With gyroscopic moments:
  $I_z\,\ddot{\psi} = -(I_{wr,\theta} - I_{wr,rot})w_{rot}\dot{\theta} - M_\delta \cos\delta + L_f F_f - L_r L_r$
- Without gyroscopic moments: $I_z\,\ddot{\psi} = L_f F_f - L_r L_r$

Final Equations of Motion

$$m\ddot{y} + mV\dot{\psi} + mh\ddot{\theta}\cos\theta - mh\theta^2 \sin\theta = F_f + F_r \tag{23}$$

$$I_z\,\ddot{\psi} = L_f F_f - L_r L_r - (I_{wr,\theta} - I_{wr,rot})w_{rot}\dot{\theta} - M_\delta \cos\delta \tag{24}$$

$$I_x\ddot{\theta} = mgh\sin\theta - mh^2\ddot{\theta}\sin^2\theta - mh^2\dot{\theta}^2\cos\theta\sin\theta -$$
$$-F_f h\cos\theta - F_r h\cos\theta + 2(I_{wf,\psi} - I_{wf,rot})w_{rot}(\dot{\psi} + \dot{\delta}\cos\beta) + M_t \tag{25}$$

$$2\,I_{wf,\psi}\ddot{\delta} = (M_\delta + M_{trail})\cos\beta - 2(I_{wf,\theta} - I_{wf,rot})w_{rot}(\dot{\theta} - \dot{\delta}\sin\beta) \tag{26}$$

The part emphasized in red corresponds to the contribution of the gyroscopic terms to the tilting, whereas the blue part is the torque provided by the tilting motor.

Linearized Equations of Motion

The linearization is punctual, around the $\theta = 0$ point. The longitudinal velocity is assumed constant, ($V_x = 8m/s$), which is one of the most constraining hypotheses, and the angle of inclination is assumed to remain close to o. To validate the use of this simplified model (dynamic simplifications and linearization), several velocities will be considered and the model will be linearized around several values.

The linearized equations using small angle approximations:

$$m\ddot{y} + mV\dot{\psi} + mh\ddot{\theta} = F_f + F_r \tag{27}$$

$$I_z\,\ddot{\psi} = L_f F_f - L_r L_r - (I_{wr,\theta} - I_{wr,rot})w_{rot}\dot{\theta} - M_\delta \tag{28}$$

$$I_x\ddot{\theta} = mgh\theta - F_f h - F_r h + +2(I_{wf,\psi} - I_{wf,rot})w_{rot}(\dot{\psi} + \dot{\delta}) + M_t \tag{29}$$

$$2\,I_{wf,\psi}\ddot{\delta} = M_\delta + M_{trail} - 2(I_{wf,\theta} - I_{wf,rot})w_{rot}\dot{\theta} \tag{30}$$

Simplifying Hypotheses

The major simplifying hypotheses which led to the production of this bicycle model, which will serve as a support for the synthesis of control laws, are listed below:

1. Only 3 degrees of freedom: the lateral position y, and the tilt and yaw angles θ,ψ.
2. The longitudinal velocity V_x is considered as an exogenous signal and not a system state variable. The model (2.22) is Linear Time-Variant (LTV), V_x being a variant parameter.
3. The linearization is obtained around the angle of inclination $\theta = 0$.
4. The vertical forces are assumed to be equal on both front wheels.
5. The stiffness (slip coefficient) C_f and the camber coefficient λ_f are assumed to be constant.
6. The effect of the suspensions is neglected
7. The wheels and the chassis are considered to be inclined to the same angle, and the point of contact of the tires do not change position during tilting.
8. The different mechanical parts moving within the vehicle, suspensions, links between different mechanical parts are not modeled.
9. The vehicle is assimilated to a mass located at the center of gravity which moves with the point of contact.

Literature Summary

To sum up, in this chapter the literature regarding tilting vehicles has been reviewed. First, the difference between active and passive systems was introduced, along with some examples of narrow three wheeler vehicles available to this day. In the field of active tilting systems, the distinction between the main control strategies was explained (DTC, STC and SDTC). Finally, using the literature, a dynamic model of the vehicle was derived.

The obtained linearized model is going to be the point of departure in the next chapter, regarding the control strategy.

3. Control Strategy

The objective of this chapter is to present the control strategy used in the PEV, based on the Direct Tilt Control (DTC). First, the lateral stability of the vehicle is studied, in order to understand the different strategies for lateral control. Then, the dynamic model is transformed into an space state model, much more easy to work with in control theory. Finally, the theory behind Linear Quadratic Regulator (LQR) and the Regulator Problem with Internal Stability (RPIS) is explained and the control gains are calculated.

Stability Study

Lateral Acceleration

The lateral acceleration a_{lat} denotes the normal acceleration induced by the curvilinear motion of the vehicle, measured in the plane XY at ground level at point D (Figure 36), and directed towards the center of rotation of the trajectory.

$$a_{lat} = \ddot{y} + V_x\dot{\psi}$$

Therefore, a_{lat} is a function of the yaw rate (and hence the steering angle) and the longitudinal speed imposed by the driver. It is independent of the vertical position (angle of inclination) of the chassis.

Perceived Lateral Acceleration

In the frame reference y', z', the acceleration at the point G decomposes into a_{per} in the y' direction and into a_z in the z' direction. The perceived lateral acceleration a_{per} is the result of the accelerations at the center of gravity of the vehicle along the y'.

$$a_{per} = a_{lat}\cos\theta + h\ddot{\theta} - g\sin\theta \tag{31}$$

The objective for the control of lateral stabilization of the vehicle and the comfort of the driver will be to impose $a_{per} = 0$. The vehicle is in lateral equilibrium if the perceived lateral acceleration is zero, i.e. $a_{per} = 0$ (**sufficient condition for lateral stability**)[41].

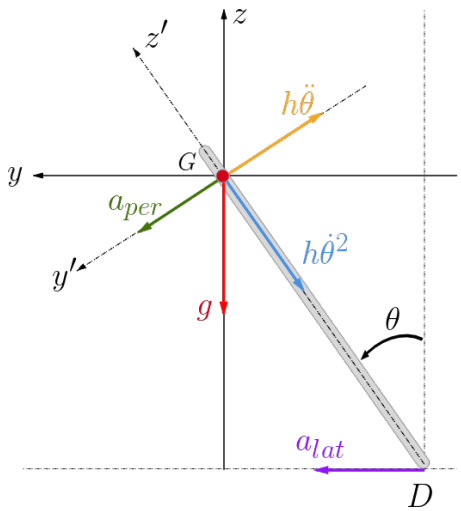


Figure 36: Forces acting on the center of gravity

[41] Lama Mourad. *Active lateral acceleration control of a narrow tilting vehicle*. Theses, Ecole des Mines de Nantes, December 2012. Available at https://tel.archives-ouvertes.fr/tel-00787310/file/Mourad_L_12_2012.pdf

The fact that a_{per} depends on a_{lat} and θ, implies that it depends on the trajectory, on the longitudinal velocity, and on the angle of inclination. However, the angle of inclination θ will be the key variable to ensure stability. We define by θ_{des} the desired position of θ which ensures $a_{per} = 0$.

Lateral Stability

In this section the dynamics of the vehicle are examined from the expression of a_{per}, in three distinct cases:

1. **Vertical vehicle in straight path**

   In this case $\theta = 0$ and $a_{lat} = 0$, therefore $a_{per} = 0$ and the problem of lateral stability does not arise, except for the cases of rejection of lateral disturbance forces (i.e. side winds), or disturbances due to an asymmetric pavement on the left and right wheels.

2. **Vertical vehicle in circular trajectory**

   When the vehicle does not tilt (i.e $\theta = 0$), then $a_{per} = a_{lat}$ (see equation (31)). If the intersection of the resultant accelerations at the center of gravity G with the plane of the ground (point x_G) is outside the base of the vehicle, then it will be unstable.

   $$a_G = a_{lat}\,\vec{j} - g\,\vec{k}$$

   On the contrary, if the point x_G recalls inside the width of the vehicle, it will be stable. Then, the maximum lateral acceleration tolerated without inclination before the overturn is:

   $$a_{lat-max} = \frac{gb}{2h} \tag{32}$$

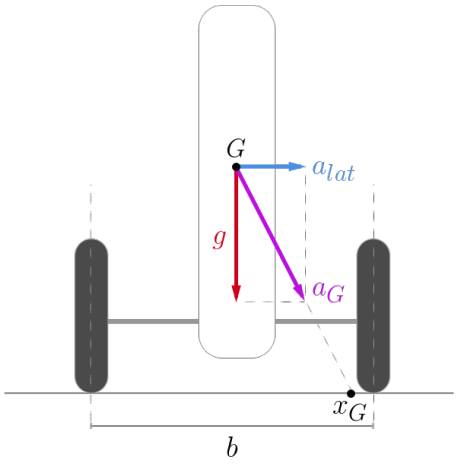


Figure 37: Forces acting on the center of gravity

3. **Inclined vehicle in circular trajectory**

   When the vehicle is inclined at a certain angle θ, its center of gravity has an offset, as well as the point x_G. The vehicle is stable as long as x_G belongs to the segment b and is, therefore, located between the two wheels of the vehicle. Let θ_n be the nominal inclination angle for which all accelerations in G are compensated, so that is solution of $a_{per} = 0$. The computation of θ_n can seem complicated, but by considering the steady state (i.e. $\ddot{\theta} = \dot{\theta} = 0$): $a_{per} = a_{lat}\cos\theta - g\sin\theta$. We then deduce the following expression:

   $$\theta_{ref} = \theta_n = \tan^{-1}\left(\frac{a_{lat}}{g}\right)$$

   This angle gives a resultant a_G colinear with the axis z' of the vehicle. Thus, the angle θ_n (solution of $a_{per} = 0$) is an angle that guarantees the lateral stability of the vehicle.

Different strategies for lateral control

The objective is to control the inclination of the vehicle so as to ensure its stability. In this way, the lateral stability of the vehicle and the comfort of passengers go hand in hand. From the sufficient condition of the stability explained above, the way to achieve this is by having null perceived acceleration at the center of gravity, $a_{per} = 0$.

Two control strategies can be followed to reach this objective:

1. The stability is ensured by a **direct angular position control**, considering the angle of inclination of reference θ_{ref} ensuring $a_{per} = 0$. This strategy has been widely adopted by researchers working on this issue. However, this strategy has some drawbacks: approximations have to be made to obtain at each moment the adapted reference angle, and its calculation potentially induces a **delay in the control**.

2. An alternative solution is to **directly control the perceived acceleration**, and to pursue the $a_{per} = 0$ objective. This is the strategy followed in this thesis.

Provided that the most readily available measurements on the vehicle are those provided by an inertial unit (IMU), the choice of direct control of a_{per} was better adapted for these sensors.

State of the Art for DTC, STC and SDTC systems

Having defined the vehicle model, explained the problem of lateral stability, and calculated the various quantities concerned (lateral acceleration, perceived acceleration) it is now possible to clearly explain the work and results obtained by different researchers.

– **Hibbard Robbin *et al.* 1996**[42], considered the problem of DTC vehicles and proposed to control the angle of inclination with the value of the reference calculated according to θ_{ref}. The lateral acceleration was approximated by $V_x\dot{\psi}$, ignoring the dependence in $\ddot{y}$

$$\theta_{ref} = \tan^{-1}\left(\frac{a_{lat}}{g}\right) = \tan^{-1}\left(\frac{\ddot{y} + V_x\dot{\psi}}{g}\right) \approx \frac{V_x\dot{\psi}}{g}$$

The control used for position feedback was a Proportional Derivative (PD) that has the error input on θ, $e = \theta - \theta_{ref}$ with a feedforward function $a_{lat} = V_x\psi$. The model used for regulator adjustment was reduced to the simplified transfer function between θ and M_t.

[42] Robin Hibbard and Dean Karnopp. Twenty first century transportation system solutions-a new type of small, relatively tall and narrow active tilting commuter vehicle. *Vehicle system dynamics*, 25(5):321–347, 1996. Available at http://dx.doi.org/10.1080/00423119608968970

– **So S. *et al.*, 1997a**[43], were interested in a system where STC and DTC actuated alternately, depending on the longitudinal speed of the vehicle.

  * A filtered PD regulator was used for the DTC system:

$$M_t = \left(K_p + \frac{K_d}{\tau_1 s + 1}\right)(\theta - \theta_{ref})$$

  * The STC strategy similarly used a PD with filtered derivative to calculate the appropriate steering angle.

  * The authors exploited the approximate relationship between a_{lat} and δ: $a_{lat} = \ddot{y} + V\dot{\psi} = (\dot{V}\, L_r\, \delta + V\, L_r\, \dot{\delta} + V^2\delta)/(L_f + L_r)$. So $\theta_{ref-STC} = a_{lat}/g = f(\delta, \dot{\delta})$ and the control signal STC is written:

$$\delta = \left(G_p + \frac{G_d s}{\tau_2 s + 1}\right)(\theta - \theta_{ref-STC})$$

  * The two controllers were first simulated separately, considering only the simplified transfer functions between δ and θ in STC and between the tilt torque M_t and θ for DTC. The proposed results showed that the perceived acceleration in DTC was much greater than in STC, and that at the same time the steering angle was significantly modified in STC;

  * Switching between STC and DTC systems was also considered, at speeds of $7.5m/s$ and $8.5m/s$ depending on a hysteresis cycle. To circumvent the jump problems with respect to the static control value, the authors proposed a variable gain for the STC, which increased gradually to optimal value.

– In their work, **So S. *et al.*, (1997)**[44] replaced the PD controller with a PI for the DTC system to solve the static error problem and minimize discontinuity during STC-DTC switching. They added an additional feedback gain as a function of θ and $\dot{\theta}$.

  The validation of this work was very basic since no stability study was available, the lateral a_{lat} and perceived a_{per} accelerations were reconstructed by their approximate mathematical expressions and were not supposed to be directly measured; The simulations were carried out on a simplified model, linearized around $\theta = 0$, considering the lateral and longitudinal dynamics.

[43] Sang-Gyun So and Dean Karnoop. Active dual mode tilt control for narrow ground vehicles. *Vehicle System Dynamics*, 27(1):19–36, 1997. Available at http://dx.doi.org/10.1080/00423119708969321

[44] Sang-Gyun So and Dean Karnopp. Switching strategies for narrow ground vehicles with dual mode automatic tilt control. *International Journal of Vehicle Design*, 18(5):518–532, 1997. Url:http://www.inderscienceonline.com/doi/abs/10.1504/IJVD.1997.062071

– At the University of Minneapolis, the researchers addressed several aspects of lateral control of narrow tilting vehicles. In their work, **Piyabongkran D. *et al.* (2004)**[45], proposed three DTC control strategies in order to control the angle of inclination at $\theta_{ref} = \tan^{-1}(V\dot{\psi}/g)$, based on the linearized lateral model.

  * The first strategy was based on a Linear Quadratic Regulator (LQR) using a simplified model whose states only corresponded to θ and $\dot{\theta}$, hence ignoring the coupling with other important variables such as the yaw rate $\dot{\psi}$. A model of the driver was implemented to correct errors on the trajectory in simulation.

  * The second strategy was based on the LQR control with a predictive aspect provided by the predictive estimation of the reference inclination angle, based on the knowledge of the future trajectory (i.e. on-board camera, GPS, etc. thus predicting the curvature of the road). The reference angle was obtained through a predictive command with a Receding Horizon Controller.

  * The third proposed control strategy was nonlinear; it proceeded by inversion of the dynamics of the inclination angle (linearization by looping), which resulted in a double integrator dynamics $M_t = \ddot{\theta}$. A PID was then used such that $M_t = (K_p + K_d s + K_i/s)(\theta - \theta_{ref})$

  The simulations showed that the nonlinear controller (strategy 3) was much more efficient than the LQ controller (strategy 2) in minimizing the tilt torque. The best performances were nevertheless obtained with the RCH controller (strategy 2), underlining the importance of the anticipation of the transient performance of the inclination control.

[45] D Piyabongkarn, T Keviczky, and R Rajamani. Active direct tilt control for stability enhancement of a narrow commuter vehicle. *International Journal of Automotive Technology*, 5(2):77–88, 2004. Available at https://pdfs.semanticscholar.org/b079/59e2ab35f552f1b06209af731e148e32a0ec.pdf

– The SDTC commands were discussed by **Kidane, S. *et al.* (2008), (2010)**[46]. The authors proposed 3 DTC strategies, and 3 STC strategies, in order to find the methodology ensuring the best switching in SDTC.

  The **DTC strategies** were all based on a PD controller, but differed in the expression of the reference θ_{ref} chosen:

  1. $\theta_{ref} = \frac{V\dot{\psi}}{g} = \frac{V^2}{gR}$
  2. $\theta_{ref} = \frac{\ddot{y}+V\dot{\psi}}{g} - \frac{2(I_{wheelrot}\, w\, \dot{\psi}/(hm)}{g}$
  3. $\theta_{ref} = k_s(\delta_{driver})$

[46] S. Kidane, L. Alexander, R. Rajamani, P. Starr, and M. Donath. A fundamental investigation of tilt control systems for narrow commuter vehicles. *Vehicle System Dynamics*, 46(4):295–322, 2008. Available at http://www.tandfonline.com/doi/abs/10.1080/00423110701352987; and S. Kidane, R. Rajamani, L. Alexander, P. J. Starr, and M. Donath. Development and experimental evaluation of a tilt stability control system for narrow commuter vehicles. *IEEE Transactions on Control Systems Technology*, 18(6):1266–1279, Nov 2010. Available at http://ieeexplore.ieee.org/document/5356230/

The first expression is the most simplified; it neglects the term of lateral slip $\ddot{y}$, which leads to a significant increase in the transient torque. The second expression is more rigorous and the term $2(I_{wheelrot}\, w\, \dot{\psi}/(mgh)$, which takes into account the gyroscopic moments at the level of the wheels, leads to a zero static error, but it is difficult to calculate. Finally, to circumvent the problem of contradictory STC and DTC control signals, the authors retained the third expression (i.e. $\theta_{des} = k_s(\delta_{driver})$), although it is very approximate and does not optimize the torque required in transient situations.

The **STC strategies** also used a PD controller, with an additional term δ_{ss}, which allowed a steering angle equal to that desired by the driver to be obtained in steady state: $\delta = K_p(\theta - \theta_{ref}) + K_d\dot{\theta} - \dot{\theta}_{ref}) + \delta_{ss}$. The three STC strategies differed in their choice of δ_{ss}:

1. The most complete proposed expression uses the value of unmeasured and therefore inaccessible variables.

2. An alternative expression performs the rewriting of expression 1. as a function of θ_{ref} and parameters to be estimated.

3. The last strategy replaces δ_{ss} by an integral term on the deviation $(\theta - \theta_{ref})$. This is the solution chosen for the SDTC command.

– The **SDTC controller** was then made up of two independent PIDs; one controller generated the motor torque M_t, and the second the steering angle δ applied to the wheels. V_o and V_f were two threshold values, $V_o < V_f$ for the longitudinal velocity. For $V < V_o$, the DTC system is active; for $V > V_f$, the STC system takes over; For $V_o < V < V_f$ a function dependent on V shared the control between DTC and STC.

The controller was validated in simulation and experimentally by **Kidane S. *et al.*, 2010**[47]. The results were acceptable, but the perceived lateral acceleration was not given, and no stability study was proposed. On the other hand, the proposed SDTC controller did not explicitly optimize the energy required for implementing this strategy.

[47] S. Kidane, R. Rajamani, L. Alexander, P. J. Starr, and M. Donath. Development and experimental evaluation of a tilt stability control system for narrow commuter vehicles. *IEEE Transactions on Control Systems Technology*, 18(6):1266–1279, Nov 2010. Available at http://ieeexplore.ieee.org/document/5356230/

– In the Netherlands, **Carver Europe** developed a three-wheel recliner called Carver. This vehicle was equipped with a Dynamic Vehicle Control system that used the DTC and STC strategies. The publications relating to this system remain evasive, probably due to the confidentiality of the project. However, several patents are filed separately concerning the use of the steering angle in the tilt controller (CR Van Den Brink *et al.*, 1999)[48], the simultaneous action of the STC and DTC systems (CR Van Den Brink And Kroonen, 2004)[49].

– At the **University of Bath**, researchers worked on the **Clever** project: a three-wheeled vehicle, in which only the chassis and front wheel were tilted. This vehicle was equipped with a DTC system, operating with hydraulic actuators. In their work, Berote *et al.*, (2008)[50] the authors focused on the analysis of the Clever dynamics, emphasizing the importance of rear wheel steering to limit the risk of over-turning of the vehicle. The authors also evoked the advantage of having the axis of rotation of the vehicle inclined (and not horizontal), which affects the inertia of the inclination movement. The DTC controller remained very basic: a PD to control the angle of inclination.

[48] CR van den Brink. Realization of high performance man wide vehicles (mwvs) with an automatic active tilting mechanism. In *Proceedings EAEC European Automotive Congress "Vehicle Systems Technology for the Next Century: Conference II–Vehicle Dynamics and Active Safety", Barcelona*, pages 41–49, 1999. Available at http://carver-technology.com/PDF/brink_EAEC_Barcelona_1999.pdf

[49] CR van den Brink and HM Kroonen. Dynamic vehicle control for enclosed narrow vehicles. In *Volume I EAEC 6th European Congress "Lightweight and small cars: The answer to Future Needs"*, pages 217–226, 1997. Available at http://www.carver-technology.nl/PDF/brink_EAEC_Cernobbio_1997.pdf

[50] Johan Berote, Auguste Van Poelgeest, Jocelyn Darling, Kevin A Edge, and Andrew Plummer. The dynamics of a three-wheeled narrow-track tilting vehicle. In *FISITA World Automotive Congress 2008*, September 2008. Paper number: F2008-SC-032. Available at http://opus.bath.ac.uk/15090/

DTC: Summary of Difficulties and Solutions

In this section the problems of DTC control are identified, and solutions are proposed to improve the performance of the controller.

Difficulties related to DTC

DTC strategies are generally based on a pursue of the angle of inclination θ, such that $\theta \rightarrow \theta_{ref}$, where θ_{ref} is given by the expression $\theta_{ref} = \tan^{-1}(a_{lat}/g)$ or even more simplified or approximate versions which have been described in detail in the previous section (i.e. $\theta_{ref} = V_x\dot{\psi}/g$, or $\theta_{ref} = k_s(\delta_{driver})$, etc.). The weak points of these strategies can be summarized as follows:

1. **High torque from the actuator on the transient phase**: the calculation of θ_{ref} from the measurement of lateral acceleration a_{lat} or yaw angle $\dot{\psi}$, induces an **intrinsic delay**: the actuator is only activated after the detection of the lateral acceleration through the accelerometer. The inclination torque must therefore compensate for the lateral acceleration already present, which inclines the vehicle towards the outside of the curve before being able to incline it in the opposite direction with the actuator.

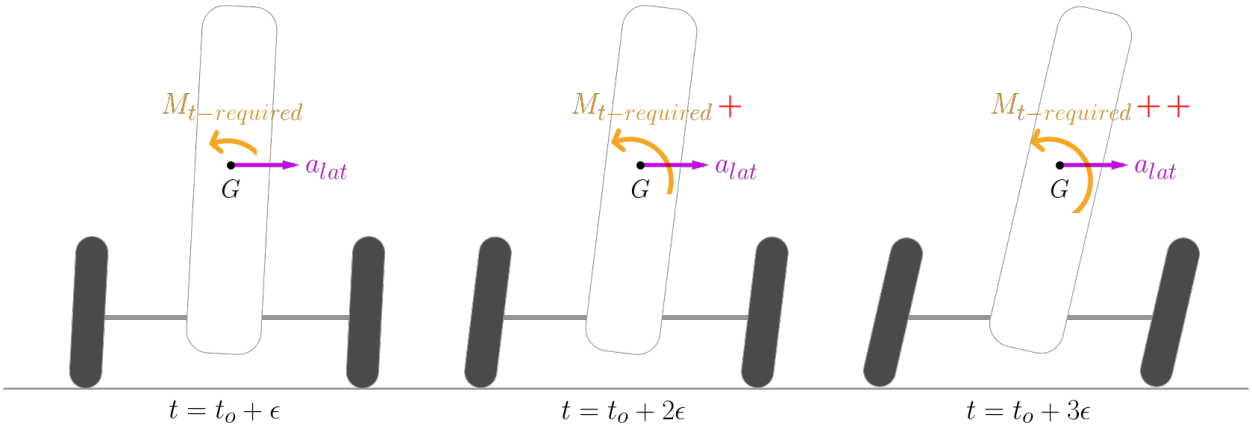


Figure 38: Transitional phase of the vehicle approaching a turn: increasing a_{lat}, inclining the vehicle outside the turn and requiring a greater M_t

The longer the M_t reaction, the greater the lateral acceleration, and the greater the required tilt torque. When other sensors that allow the trajectory prediction are installed on the vehicle (camera), this anticipation makes it possible to reduce the tilting torque significantly (Piyabongkran D. and Al., 2004)[51]. In addition, the lateral acceleration ($a_{lat} = \ddot{y} + V\dot{\psi}\cos\beta$) is proportional to the speed. Thus, the higher the speed, the greater the lateral acceleration and the greater the torque M_t required.

[51] D Piyabongkarn, T Keviczky, and R Rajamani. Active direct tilt control for stability enhancement of a narrow commuter vehicle. *International Journal of Automotive Technology*, 5(2):77–88, 2004. Available at https://pdfs.semanticscholar.org/b079/59e2ab35f552f1b06209af731e148e32a0ec.pdf

2. **Static error**: Given the way in which the reference is calculated, $\theta = \theta_{ref}$ does not mean that a_{per} is strictly null. Consequently, the actuator must permanently supply a non-zero torque to maintain the inclination of the vehicle at θ_{ref}.

Proposed solutions

The two main solutions to improve the operation of the DTC systems are: A) Using the steering angle input from the driver as a regulator and B) Directly controlling the perceived lateral acceleration.

1. **Reduction of the actuator torque in DTC**

   When driving of a motorcycle, the rider tilts it as soon as the turn begins, before the appearance of the lateral acceleration. Consequently the effort required for the inclination is relatively small; and this is also the case for a vehicle whose path is known in advance. It is, thus, essential to initiate the inclination as soon as possible.

   The first measurable signal containing information about the near trajectory is the **angle (or even the torque) of steering**. This signal must be used in order to initiate the inclination of the vehicle before the perception of lateral acceleration.

   The steering angle is considered as a **disturbance** which induces an increase in the perceived lateral acceleration, such as to destabilize the vehicle. In order to be able to use this signal in the controller, a control methodology is chosen to be able to take into account this knowledge of the environment.

2. **Cancellation of the static error and improvement of the DTC performance**

   The lateral stability is obtained by the appropriate inclination of the vehicle. It is therefore intuitive to seek to control the angle of inclination $\theta \rightarrow \theta_{ref}$ by calculating the necessary torque to achieve an inclination coherent with the terminal objective: $a_{per} = 0$. To reduce the problem of the approximation and to reduce the risks of instability (θ_{ref} is deduced from the state variables) the **direct control of the perceived lateral acceleration** a_{per} is proposed.[52]

   Therefore, the objective variable is now the $a_{per} = 0$ rather than $\theta = \theta_{ref}$. In addition, in order to ensure the robust asymptotic cancellation of the static error, the integral of a_{per} (a^I_{per}) will be taken into account. This approach is much more interesting since a_{per} is measured by the inertial motion unit (IMU).

[52] L. Mourad, F. Claveau, and P. Chevrel. Design of a two dof gain scheduled frequency shaped lq controller for narrow tilting vehicles. In *2012 American Control Conference (ACC)*, pages 6739–6744, June 2012. Available at http://ieeexplore.ieee.org/abstract/document/6315042/

Design of the DTC Controller

Dynamic Model

In the previous chapter, the linearized dynamic model was obtained:

$$m\ddot{y} + mV\dot{\psi} + mh\ddot{\theta} = F_f + F_r \quad (33)$$

$$I_z\,\ddot{\psi} = L_f F_f - L_r L_r - (I_{wr,\theta} - I_{wr,rot})w_{rot}\dot{\theta} - M_\delta \quad (34)$$

$$I_x\ddot{\theta} = mgh\theta - F_f h - F_r h + +2(I_{wf,\psi} - I_{wf,rot})w_{rot}(\dot{\psi} + \dot{\delta}) + M_t \quad (35)$$

$$2\,I_{wf,\psi}\ddot{\delta} = M_\delta + M_{trail} - 2(I_{wf,\theta} - I_{wf,rot})w_{rot}\dot{\theta} \quad (36)$$

$$F_f = 2C_f\left(\delta - \frac{\dot{y} + L_f\dot{\psi}}{V}\right) + 2\lambda_f\theta$$

$$F_r = C_r\left(-\frac{\dot{y} - L_r\dot{\psi}}{V}\right) + \lambda_r\theta$$

The inputs to this model are the steering angle δ and the tilting torque M_t.

Methodological Approach

The general idea consists in formalizing the problem through the construction of a generic model, called standard, allowing the reformulation of the problem of control in an optimization problem H_2[53]. This model $P(S)$ is structured and consists of the aggregation of:

- **Plant**: A model of the system to be monitored, displaying disturbance and control signals and well as the output variables to be controlled.
- **Disturbance**: A model of the environment of the system to be controlled, generating the exogenous signals such as perturbations and setpoints.
- **Output**: A model of the quantities to be regulated, including weights constituting adjustment coefficients.

The associated H_2 is considered in the generic form, which consists in determining K_s stabilizing the process model and minimizing: $\|P(s)\,K(s)\|_2$

[53] Kemin Zhou, John C. Doyle, and Keith Glover. *Robust and Optimal Control*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1996. ISBN:0-13-456567-3

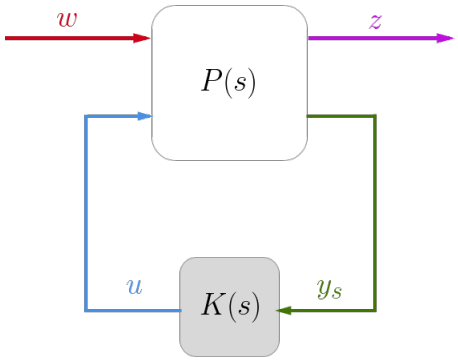


Figure 39: Standard Problem

State Space Model

The obtained model of the form described by equations (33-36) is reduced to its controllable and observable part which has 4 states $(\dot{y}, \dot{\psi}, \theta, \dot{\theta})$. The Linear Parameter Variant (LTV) model is parameterized by the longitudinal speed V of the vehicle:

$$\begin{cases} \dot{x} = A\,x + B_u\,u + B_d\,d \\ y = C\,x + D_u\,u + D_d\,d \end{cases} \tag{37}$$

with $x^T = \begin{bmatrix} \dot{y} & \dot{\psi} & \theta & \dot{\theta} \end{bmatrix}^T \in \Re^4$, $u = M_t$ and $d = \delta$

$$A = \begin{bmatrix} \frac{1}{V}\left(-\frac{a}{m} - \frac{h^2 a}{I_x}\right) & \frac{1}{V}\left(-\frac{b}{m} - \frac{h^2 b}{I_x}\right) - V & (2\lambda_f + \lambda_r)\left(\frac{1}{m} + \frac{h^2}{I_x}\right) - \frac{mgh^2}{I_x} & 0 \\ -\frac{b}{V I_z} & -\frac{2C_f L_f^2 + C_r L_r^2}{V I_z} & \frac{2\lambda_f L_f - \lambda_r L_r}{I_z} & 0 \\ 0 & 0 & 0 & 1 \\ \frac{h\,a}{V I_x} & \frac{h\,b}{V I_x} & \frac{mgh - h(2\lambda_f + \lambda_r}{I_x} & 0 \end{bmatrix}$$

$$B_d = B_\delta = \begin{bmatrix} 2C_f\left(\frac{1}{m} + \frac{h^2}{I_x} & \frac{2C_f L_f}{I_z} & 0 & -\frac{2C_f h}{I_x} \end{bmatrix}^T,$$

$$B_u = B_{M_t} = \begin{bmatrix} -\frac{h}{I_x} & 0 & 0 & \frac{1}{I_x} \end{bmatrix}^T,\ a = 2\,C_f + C_r,\ b = 2\,C_f L_f - C_r L_r$$

The model becomes an Linear Time Invariant (LTI) model when a constant longitudinal velocity V is considered. The vector x denotes the states , y the measured the outputs, u the control input and d is considered as a disturbing exogenous signal[54].

[54] Rensselaer Polytechnic Institute Dr. Kevin Craig, Professor of Mechanical Engineering. *Disturbance Response Notes*. Available at http://studylib.net/doc/18342577/disturbance-response

Estimation of $\dot{y}$

The PEV will include a Inertial Measurement Unit (IMU), which will provide the state values θ, $\dot{\theta}$, and $\dot{\psi}$, but not $\dot{y}$. The IMU will also give the value for the perceived lateral acceleration a_{per}. Note that this measure was not used in the previous strategy, nor in the classical strategy of controlling θ such that $\theta = \theta_{des}$ steering angle δ and its derivative $\dot{\delta}$ will be measured.

The state signal $\dot{y}$ can be estimated from the measured signals: $y_p = \begin{bmatrix} \dot{\psi} & \theta & \dot{\theta} & a_{per} \end{bmatrix}^T$, according to:

$$\hat{\dot{y}} = (a_{11} + h a_{41})^{-1}\Big[a_{per} - (a_{12} + h a_{42} + V_x)\dot{\psi} - (a_{13} + h a_{43} - g)\theta \\ -(a_{14} + h a_{44})\dot{\theta} - (b_{\delta 1} + h b_{\delta 4})\delta - (b_{u1} + h b_{u4})M_t\Big]$$

where terms a_{ij}, $b_{\delta i}$ and b_{ui} are coefficients of matrices A, B_δ and B_u respectively.

Expressing a_{per} as function of x

Lateral stability of the vehicle is obtained when the sum of lateral forces and torques at its center of gravity is zero ($a_{per} = 0$).

$$a_{per} = a_{lat}\cos\theta + h\ddot{\theta} - g\sin\theta \quad \text{with} \quad a_{lat} = \ddot{y} + V\dot{\psi}$$

The a_{per} is a measured signal, but is not part of the state vector. Using small angle approximations, it can be expressed as a function of the state vector as follows:

$$a_{per} \approx \ddot{y} + V\dot{\psi} + h\ddot{\theta} - g\theta = a_{per}^{lin}$$

$$a_{per}^{lin} = \begin{bmatrix} 0 & V & -g & 0 \end{bmatrix} x + \begin{bmatrix} 1 & 0 & 0 & h \end{bmatrix} \dot{x} = G_1\, x + G_2\, \dot{x}$$

Replacing $\dot{x}$ by $A\,x + B_u\,u + B_d\,d$ leads to:

$$a_{per}^{lin} = G_1\, x + G_2(A\,x + B_u\,u + B_d\,d) = G\,x + H_u\,u + H_d\,d$$

with $G = G_1 + G_2\,A$, $H_u = G_2\,B_u$ and $H_d = G_2\,B_d$. The output vector is defined as $y = \begin{bmatrix} a_{per} & \dot{\psi} & \theta & \dot{\theta} & \delta \end{bmatrix}^T$ Therefore:

$$C = \begin{bmatrix} G \\ \begin{bmatrix} 0_{3x1} & I_3 \end{bmatrix} \\ 0_{1x4} \end{bmatrix},\ D_u = \begin{bmatrix} H_u \\ 0_{4x1} \end{bmatrix},\ D_d = \begin{bmatrix} H_d \\ 0_{3x1} \\ 1 \end{bmatrix}$$

Regulator Problem with Internal Stability [55]

The fundamental control objective to ensure the lateral stability is to solve the regulation problem $a_{per} = 0$. The problem formulation is recast as a standard regulator problem with internal stability (RPIS) where δ is considered as a disturbance, and its effect on the perceived lateral acceleration a_{per} should be canceled by the tilt torque M_t.

[55] L. Mourad, F. Claveau, and P. Chevrel. Design of a two dof gain scheduled frequency shaped lq controller for narrow tilting vehicles. In *2012 American Control Conference (ACC)*, pages 6739–6744, June 2012. Available at http://ieeexplore.ieee.org/abstract/document/6315042/

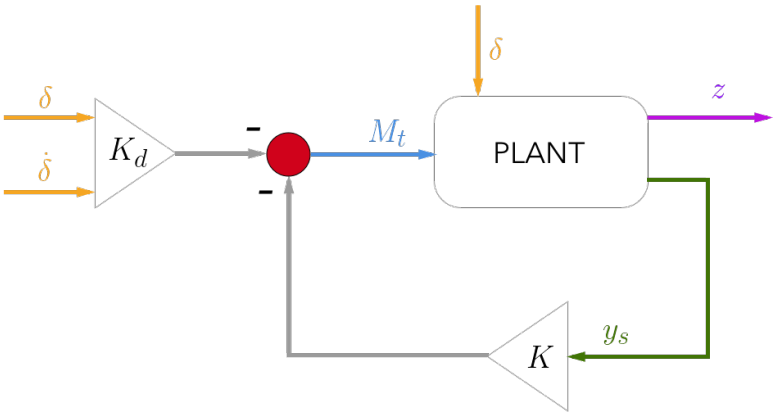


Figure 40: Control diagram: Feedforward and feedback loops

Plant Model

$$\begin{cases} \dot{x} = A\,x + B_u\,u + B_d\,d \\ y = C\,x + D_u\,u + D_d\,d \\ z = G\,x + H_u\,u + H_d\,d \end{cases} \tag{38}$$

with $G = \begin{bmatrix} -a/V\,m & -b/V\,m & (2\lambda_f + \lambda_r)/m - g & 0 \end{bmatrix}$

$H_u = 0, H_d = 2C_f/m$

Disturbance Model

$$\begin{cases} \dot{x_e} = A_e\,x + B_e\,w \\ d = C_e\,x_e \end{cases} \tag{39}$$

with $x_e^T = \begin{bmatrix} \delta & \dot{\delta} \end{bmatrix}^T$, $C_e = \begin{bmatrix} 1 & 0 \end{bmatrix}$, $A_e = \begin{bmatrix} 0 & 1 \\ -\tau\alpha & -(\tau+\alpha) \end{bmatrix}$, $B_e = \begin{bmatrix} 0 \\ \beta \end{bmatrix}$

where w is an impulse signal, α, τ and β determine the time constants and amplitude of the 2^{nd} order signal. This model is critical to obtain good performances in transient phases, while the feedback takes care of the static ones and guarantees robust regulation.

Standard Model

$$\begin{bmatrix} \dot{x} \\ \dot{x_e} \end{bmatrix} = \begin{bmatrix} A & B_d\,C_e \\ 0 & A_e \end{bmatrix} \begin{bmatrix} x \\ x_e \end{bmatrix} + \begin{bmatrix} B_u \\ 0 \end{bmatrix} u + \begin{bmatrix} 0 \\ B_e \end{bmatrix} w$$

$$e = z = \begin{bmatrix} G & H_d\,C_e \end{bmatrix} \begin{bmatrix} x \\ x_e \end{bmatrix} + H_u\,u$$

$$y = \begin{bmatrix} x \\ x_e \end{bmatrix} = \begin{bmatrix} \dot{y} & \dot{\psi} & \theta & \dot{\theta} & \delta & \dot{\delta} \end{bmatrix}^T$$

Solution for RPIS

Assuming that the plant model is stabilizable, a solution for the RPIS problem exists in the feedback form if the unique solution (T_a, F_a) of the occultation equations exists:

$$-A\,T_a + T_a\,A_e + B_d\,C_e = -B_u\,F_a \tag{40}$$

$$-G\,T_a + H_d\,C_e = H_u\,F_a \tag{41}$$

If $\exists\, T_a, F_a \rightarrow u_{ref} = -F_a\,x_e$ and $x_{ref} = -T_a\,x_e$ is the unique reference trajectory for the plant model that satisfies the control objective $e = a_{per} = 0$. Therefore this trajectory satisfies the dynamic of the system:

$$\begin{cases} \dot{x}_{ref} = A\,x_{ref} + B_u\,u_{ref} + B_d\,d \\ z_{ref} = G\,x_{ref} + H_u\,u_{ref} + H_d\,d \end{cases} \tag{42}$$

Defining $(\tilde{u}, \tilde{x}, e) = (u - u_{ref}, x - x_{ref}, z - z_{ref})$ the difference between the actual and desired values of the current control input, states and controlled output.

The first step is to compute the feedback gain K, designed such that $\tilde{u} = -K\tilde{x}$ guarantees $\tilde{x} \to 0$ and $e \to 0$ (stabilization of the plant). The second step a feedforward gain is designed in order for (u, x) to follow the reference trajectory u_{ref}, x_{ref} in an optimal way.

Feedback Gains

For the calculation of the feedback gains, the Linear Quadratic Regulator (LQR)[56,57,58] has been chosen based on the good robustness properties, few tuning parameters and the explicit minimization of the energy of control signals required to follow the reference.

The LQR consists on finding the state feedback $\tilde{u} = -K\tilde{x}$ that minimizes the criteria:

$$J_{LQR} = \int_0^\infty \left(z^T\, Q\, z + u^T\, R\, u\right) dt = \int_0^\infty \left(x^T\, Q_x\, x + u^T\, R\, u + 2x^T\, N_{xu}\, u\right) dt$$

The weighting matrices Q and R must be positive and will balance the minimization of the state vector or the control signal.

$$Q_x = G^T\, Q\, G$$

$$R_u = H_u^T\, Q\, H_u + R$$

$$N_{ux} = G^T\, Q\, H_u$$

The solution is given by:

$$K = -R_u^{-1}\, B_u^T\, P$$

Where P is the unique solution of the Riccati equation applied to the plant model:

$$A^T\, P + P\, A - P\, B_u\, R_u^{-1}\, B_u^T\, P + Q_x = 0$$

Feedforward Gains

Solving the Sylvester equations in (42) and using the feedback gains K, the feedforward gain is calculated as $K_d = F_a + K\, T_a$.

$$\begin{aligned} \tilde{u} = -K\tilde{x} \quad \to \quad & u - u_{ref} = -K\,(x - x_{ref}) \\ & u + F_a\, x_e = -K(x + T_a\, x_e \\ & u = -K\, x - (F_a + K\, T_a) x_e = -K\, x - K_d\, x_e \end{aligned}$$

[56] University of Stuttgart Prof. Dr. Carsten Scherer. *LQ Optimal Control*, 2013. Available at http://www.mathematik.uni-stuttgart.de/studium/infomat/mst/linearekontrolltheorie/folien/lec5.pdf

[57] Åbo Akademi University Professor Hannu T. Toivonen, Department of Information Technologies. *The H2-optimal control problem*, 2000. Available at http://users.abo.fi/htoivone/courses/robust/rob3.pdf

[58] W Murray Wonham. *Linear multivariable control a geometric approach; 2nd ed.* Applications of Mathematics. Springer, New York, 1979. Available at http://cds.cern.ch/record/1614618

Extended model with the integral of a_{per}: a_{per}^I

In order to avoid any static error, the integrated value of the lateral perceived acceleration will be controlled, and its value will be added to the state vector. In fact, it is unavoidable to have some model errors due to parameters uncertainty, neglected dynamics or linearization of the model.

The equation to add to the model is very simple:

$$a_{per}^I = a_{per} = G\dot{x} + H_u\,u + H_d\,d$$

The augmented state is then $x_i = \begin{bmatrix} \dot{y} & \dot{\psi} & \theta & \dot{\theta} & a_{per}^I \end{bmatrix}$, and the new system:

$$\begin{cases} \dot{x_i} = A\,x_i + B_{iu}\,u + B_{id}\,d \\ z = a_{per}^I = G_i\,x_i + H_{iu}\,u + H_{id}\,d \end{cases} \tag{43}$$

$$A_i = \begin{bmatrix} A & 0_{4x1} \\ G & 01x1 \end{bmatrix}, \quad B_{iu} = \begin{bmatrix} B_u \\ H_u \end{bmatrix}, \quad B_{id} = \begin{bmatrix} B_d \\ H_d \end{bmatrix}$$
$$G_i = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \quad H_{iu} = 0, \quad H_{id} = 0$$

Extended Standard Model

$$\begin{bmatrix} \dot{x_i} \\ \dot{x_e} \end{bmatrix} = \begin{bmatrix} A_i & B_{id}\,C_e \\ 0_{2x5} & A_e \end{bmatrix} \begin{bmatrix} x_i \\ x_e \end{bmatrix} + \begin{bmatrix} B_{iu} \\ 0_{2x1} \end{bmatrix} u + \begin{bmatrix} 0_{2x1} \\ B_e \end{bmatrix} w$$

$$e = z = a_{per}^I = \begin{bmatrix} G_i & H_{id}\,C_e \end{bmatrix} \begin{bmatrix} x_i \\ x_e \end{bmatrix} + H_{iu}\,u$$

$$y = \begin{bmatrix} x \\ x_e \end{bmatrix} = \begin{bmatrix} \dot{y} & \dot{\psi} & \theta & \dot{\theta} & a_{per}^I & \delta & \dot{\delta} \end{bmatrix}^T$$

Feedback Gains

The Linear Quadratic Regulator is adapted to the extended model:

$$J_{LQR} = \int_0^\infty \left(z^T\,Q\,z + u^T\,R\,u\right)dt = \int_0^\infty \left(x^T\,Q_x\,x + u^T\,R\,u + 2x^T\,N_{xu}\,u\right)dt$$

$$Q_x = G_i^T\,Q\,G_i \qquad R_u = H_{iu}^T\,Q\,H_{iu} + R \qquad N_{ux} = G_i^T\,Q\,H_{iu}$$

The solution is given by:

$$K = -R_u^{-1}\,B_{iu}^T\,P$$

Where P is the unique solution of the Riccati equation applied to the plant model:

$$A_i^T\,P + P\,A_i - P\,B_{iu}\,R_u^{-1}\,B_{iu}^T\,P + Q_x = 0$$

Feedforward Gains

The Sylvester equations in (42) need to be adapted to the new model:

$$-A_i\, T_a + T_a\, A_e + B_{id}\, C_e = -B_{iu}\, F_a \tag{44}$$

$$-G_i\, T_a + H_{id}\, C_e = H_{iu}\, F_a \tag{45}$$

By solving these equations and using the feedback gains K, the feedforward gain is calculated as $K_d = F_a + K\, T_a$.

Simple Method Gains Calculation

A straightforward way of calculating the gains is to solve the Riccati and the Sylvester equations as it is indicated in (Friedland, Chapter 9.6)[59] book:

[59] Bernard Friedland. *Control System Design - An Introduction to State-Space Methods*. Dover Publications, 1986. Chapter 9. Linear, Quadratic Optimum Control. Available at http://app.knovel.com/hotlink/toc/id:kpCSDAISS1/control-system-design/control-system-design ISBN: 978-0-486-44278-5

– Riccati eq: $M_1\, A_i + A_i^T\, M_1 - M_1\, B_{iu}\, R_u^{-1} B_{iu}^T M_1 + Q_x = 0$

– Slyvester eq: $M_2\, A_e + (A_i^T - M_1\, B_{iu}\, R_u^{-1}\, B_{iu}^T) M_2 + M_1\, B_{id}\, C_e = 0$

Then, the gains can be easily calculated:

- Feedback gains: $K = R_u^{-1}\, B_{iu}^T\, M_1$
- Feedforward gains: $K_d = R_u^{-1}\, B_{iu}^T\, M_2$

Derivation of the Gains

As was discussed previously, the state variable $\dot{y}$ is not measured and can only be estimated as:

$$\hat{\dot{y}} = (a_{11} + h a_{41})^{-1} \Big[a_{per} - (a_{12} + h a_{42} + V_x)\dot{\psi} - (a_{13} + h a_{43} - g)\theta \\ -(a_{14} + h a_{44})\dot{\theta} - (b_{\delta 1} + h b_{\delta 4})\delta - (b_{u1} + h b_{u4}) M_t \Big]$$

To calculate the real gains on the controller, a simple but tedious derivation has to be carry out:

$$u = -K\, x_i - K_e\, x_e$$

$$u = -K_{\dot{y}}\, \dot{y} - K_{\dot{\psi}}\, \dot{\psi} - K_\theta\, \theta - K_{\dot{\theta}}\, \dot{\theta} - K_{a_{per}^I}\, a_{per}^I - K_\delta\, \delta - K_{\dot{\delta}}\, \dot{\delta}$$

$$u = -\frac{K_{\dot{y}}}{a_{11}+ha_{41}}\,a_{per} - \Big(K_{\dot{\psi}} - K_{\dot{y}}\frac{a_{12}+ha_{42}+V_x}{a_{11}+ha_{41}}\Big)\dot{\psi} - \Big(K_{\theta} - K_{\dot{y}}\frac{a_{13}+ha_{43}-g}{a_{11}+ha_{41}}\Big)\theta - \Big(K_{\dot{\theta}} - K_{\dot{y}}\frac{a_{14}+ha_{44}}{a_{11}+ha_{41}}\Big)\dot{\theta} -$$

$$- K_{a^I_{per}}\,a^I_{per} - \Big(K_{\delta} - K_{\dot{y}}\frac{b_{\delta 1}+hb_{\delta 4}}{a_{11}+ha_{41}}\Big)\delta - K_{\dot{\delta}}\,\dot{\delta} - K_{\dot{y}}\,\frac{b_{u1}+hb_{u4}}{a_{11}+ha_{41}}\,u$$

$$u = \frac{1}{1+K_{\dot{y}}\,\frac{b_{u1}+hb_{u4}}{a_{11}+ha_{41}}}\Big[-\frac{K_{\dot{y}}}{a_{11}+ha_{41}}\,a_{per} - \Big(K_{\dot{\psi}} - K_{\dot{y}}\frac{a_{12}+ha_{42}+V_x}{a_{11}+ha_{41}}\Big)\dot{\psi} - \Big(K_{\theta} - K_{\dot{y}}\frac{a_{13}+ha_{43}-g}{a_{11}+ha_{41}}\Big)\theta -$$

$$- \Big(K_{\dot{\theta}} - K_{\dot{y}}\frac{a_{14}+ha_{44}}{a_{11}+ha_{41}}\Big)\dot{\theta} - K_{a^I_{per}}\,a^I_{per} - \Big(K_{\delta} - K_{\dot{y}}\frac{b_{\delta 1}+hb_{\delta 4}}{a_{11}+ha_{41}}\Big)\delta - K_{\dot{\delta}}\,\dot{\delta}\Big]$$

$$u = \frac{1}{a_{11}+ha_{41}+K_{\dot{y}}\,(b_{u1}+hb_{u4})}\Big[-K_{\dot{y}}\,a_{per} - \big[K_{\dot{\psi}}(a_{11}+ha_{41}) - K_{\dot{y}}(a_{12}+ha_{42}+V_x)\big]\dot{\psi} -$$

$$- \big[K_{\theta}(a_{11}+ha_{41}) - K_{\dot{y}}(a_{13}+ha_{43}-g)\big]\theta -$$

$$- \big[K_{\dot{\theta}}(a_{11}+ha_{41}) - K_{\dot{y}}(a_{14}+ha_{44})\big]\dot{\theta} - K_{a^I_{per}}(a_{11}+ha_{41})\,a^I_{per} -$$

$$- \big[K_{\delta}(a_{11}+ha_{41}) - K_{\dot{y}}(b_{\delta 1}+hb_{\delta 4})\big]\delta - K_{\dot{\delta}}(a_{11}+ha_{41})\,\dot{\delta}\Big]$$

With this estimation all the signals are measured and have a gain:

$$K'_{a_{per}} = \frac{K_{\dot{y}}}{a_{11}+ha_{41}+K_{\dot{y}}\,(b_{u1}+hb_{u4})} \tag{46}$$

$$K'_{\dot{\psi}} = \frac{K_{\dot{\psi}}(a_{11}+ha_{41}) - K_{\dot{y}}(a_{12}+ha_{42}+V_x)}{a_{11}+ha_{41}+K_{\dot{y}}\,(b_{u1}+hb_{u4})} \tag{47}$$

$$K'_{\theta} = \frac{K_{\theta}(a_{11}+ha_{41}) - K_{\dot{y}}(a_{13}+ha_{43}-g)}{a_{11}+ha_{41}+K_{\dot{y}}\,(b_{u1}+hb_{u4})} \tag{48}$$

$$K'_{\dot{\theta}} = \frac{K_{\dot{\theta}}(a_{11}+ha_{41}) - K_{\dot{y}}(a_{14}+ha_{44})}{a_{11}+ha_{41}+K_{\dot{y}}\,(b_{u1}+hb_{u4})} \tag{49}$$

$$K'_{a^I_{per}} = \frac{K_{a^I_{per}}(a_{11}+ha_{41})}{a_{11}+ha_{41}+K_{\dot{y}}\,(b_{u1}+hb_{u4})} \tag{50}$$

$$K'_{\delta} = \frac{K_{\delta}(a_{11}+ha_{41}) - K_{\dot{y}}(b_{\delta 1}+hb_{\delta 4})}{a_{11}+ha_{41}+K_{\dot{y}}\,(b_{u1}+hb_{u4})} \tag{51}$$

$$K'_{\dot{\delta}} = \frac{K_{\dot{\delta}}(a_{11}+ha_{41})}{a_{11}+ha_{41}+K_{\dot{y}}\,(b_{u1}+hb_{u4})} \tag{52}$$

$$K' = \begin{bmatrix} K'_{a_{per}} & K'_{\dot{\psi}} & K'_{\theta} & K'_{\dot{\theta}} & K'_{a^I_{per}} & K'_{\delta} & K'_{\dot{\delta}} \end{bmatrix}$$

Gain Scheduled Controller

The previous regulator was synthesized by considering a constant value of the longitudinal speed V_x. But the state space model is LPV and the matrices A and C depend on V or $1/V$. Thus, the proposed controller is not valid for the whole speed range. In this section, a controller that depends on the longitudinal speed of the vehicle is proposed.

The approach adopted is pragmatic, and proceeds by interpolation of the LTI regulators obtained for a discrete set of longitudinal speeds. It thus guarantees a good performance at every operating point. Stability guarantees will also be provided by a posteriori analysis.

The first step is the observation of the gain variation of this regulator as a function of velocity for $V \in 2, 3, ...18\, m/s$. A regulator is synthesized for each value of V and the variation of the gains can be approximated by a function of the form:

$$K(V) = K_c + K_V\, V + K_{1/V)}$$

Dependent on V and $1/V$, with K_c, K_V and $K_{1/V}$ the constants to be identified. These values can be easily deduced from the least squares resolution of the following system of equations:

$$\underbrace{\begin{pmatrix} 1 & V_1 & 1/V_1 \\ 1 & V_2 & 1/V_2 \\ ... & ... & ... \\ 1 & V_{17} & 1/V_{17} \end{pmatrix}}_{M} \begin{pmatrix} K_c \\ K_V \\ K_{1/V} \end{pmatrix} = \underbrace{\begin{pmatrix} K'_{V_1} \\ K'_{V_2} \\ ... \\ K'_{V_{17}} \end{pmatrix}}_{K_M} \Rightarrow \begin{pmatrix} K_c \\ K_V \\ K_{1/V} \end{pmatrix} = (M^T M)^{-1} M^T K_M$$

Control Strategy Summary

In this chapter the control strategy has been completely derived and developed. The process of calculating the gains can be resumed in few steps:

1. To obtain or measure the **parameters to complete the dynamic model**. This includes the vehicle and the wheels variables:

$$m \;\; L_f \;\; L_r \;\; I_x \;\; I_z \;\; C_f \;\; C_r \;\; \lambda_f \;\; \lambda_r \;\; ....$$

2. To suppose a **constant** V **for the model**, so that $V \in 2, 3, ...18\, m/s$

3. To generate the **matrices of the state space** model:

$$A \;\; B_u \;\; B_d \;\; C \;\; D_u \;\; D_d \;\; G \;\; H_u \;\; H_d \;\; A_e \;\; B_e \;\; C_e \;\; ...$$

4. **Feedback gains**: To solve the Riccati equation for the selected model and calculate the gains:

$$M_1\, A_i + A_i^T\, M_1 - M_1\, B_{iu}\, R_u^{-1} B_{iu}^T M_1 + Q_x = 0$$

$$K = R_u^{-1}\, B_{iu}^T\, M_1$$

5. **Feedforward gains**: To solve the Slyvester equation for the selected model and calculate the gains:

$$M_2\, A_e + (A_i^T - M_1\, B_{iu}\, R_u^{-1}\, B_{iu}^T) M_2 + M_1\, B_{id}\, C_e = 0$$

$$K_d = R_u^{-1}\, B_{iu}^T\, M_2$$

6. **Gain Scheduling**: To repeat the steps 2 to 5 with all the speed range and calculate the constants K_c, K_V and $K_{1/V}$ with:

$$\begin{pmatrix} K_c \\ K_V \\ K_{1/V} \end{pmatrix} = (M^T\, M)^{-1}\, M^T\, K_M$$

4. Small Scale Prototype

As a first step towards the design and fabrication of a tilting PEV, an small scale prototype was built. In this chapter the design and fabrication process regarding this small vehicle, also called **miniPEV**, will be explained.

Concept - Deploy or Die

In sight of the Media Lab's motto '**deploy or die**'[60], the objective of this thesis would be to deploy a real scale vehicle with a tilting mechanism. However, the process to reach that goal needs a lot of iterations, try and errors, and a small scale prototype is a good starting point.

[60] Media Lab Director's new motto. Available at https://slice.mit.edu/2014/07/29/deploy-or-die-media-lab-directors-new-motto/

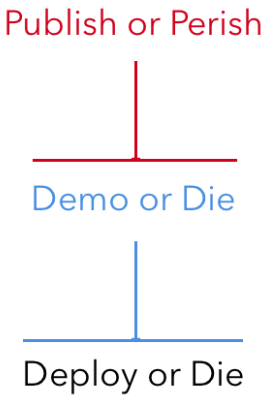


Figure 41: Evolution of the philosophy of innovation at the Media Lab

A small moving tilting prototype – **miniPEV**– is helpful in a lot of different ways. It is much easier to show your ideas and the story behind narrow tilting vehicles to people who are not familiarized with this kind of vehicles. As a matter of fact, the miniPEV was shown in the Fall Member Event at the Media Lab. The feedback received from the sponsors, students, and people from the MIT community was really helpful to understand better the problem and its opportunities in the mobility sector.

The miniPEV[61] is a simple tilting 'toy' car make out of wood. It is approximately one quarter of the scale of the PEV, but as a concept, it represents all the features that a full scale vehicle has. It is propulsed by a DC motor, and two servomotors move the two front steering wheels and the tilting mechanism. It also has a battery to power all the electronics and the motors. In order to simulate the same inputs to the control system, a Inertial Measurement Unit (IMU) was also installed and used to tilt the vehicle in the curves. The details regarding the components will be explained later in this chapter.

[61] miniPEV video. Available at https://vimeo.com/188999255

Design

The design on the miniPEV was based on a previous small prototype by Michael Lin, head of the PEV project at Changing Places group. In order to fabricate the prototype as fast as possible, the miniPEV was built with plywood sheets of $3mm$, that were cut in the laser cutting machine. Then, gluing the layers together gives result to the final 3D model. In this way, the prototype is fabricated very fast and the possibilities of making changes is high as well.

The model is composed of different parts. In the rear part there is a seat, and the wheel is propulsed from the hub. The electronics and the battery are placed in the center, with the aim of keeping the model as balanced as possible and having a low center of gravity. Due to the limitations of power, this design decision was critical for the proper motion of the model. The front part consist of two steered wheels with their steering system and a suspension that allows the tilting of the vehicle.

Tilting Mechanisms

This section focuses on explaining the tilting mechanism chosen for the miniPEV, as an introduction to the more detailed simulations carried out for the full scale PEV.

The different tilting mechanisms that previous tilting vehicles have used can be summarized in two categories:

1. **Hydraulic actuators**: located in the suspension or in a fixed part of the vehicle, they can incline the vehicle by extending one of the two actuators. The size and power provided by these actuators exceed the requirements and the space available in the PEV, as well as in the miniPEV. This strategy has been used in car-like vehicles rather than in bike-like vehicles.

2. **Rotary motor**: being much compact, a high torque motor can supply the necessary M_t to lean the vehicle when necessary. Here the possibilities of actuating the vehicle expand, but the design remains basic. Attached to the motor there is a arm that connects to the suspension arms, meaning that the rotation of the motor reduces and enlargers the distance from the body to the suspension, hence tilting the vehicle and the wheels. This second option is the one that is going to be explored next, due to the different possibilities that comprises.

Rotary Motor Tilting Mechanisms

If a rotary motor is used, the first question that arises is: What part of the vehicle would we like to tilt? The answer falls on selecting the parts that we do not want to lean. In a tadpole vehicle like PEV with 2 front wheels and 1 rear wheel (the contrary will be a delta vehicle - 1F2R), the part that can maintain vertical by itself is the front suspension and the wheels. In this way, we could separate the whole front part from the rest of the frame (handle bar, cover, seat, rear wheel...), implying that the tilting part would be the body of the vehicle. This is indeed the alternative **A** (Figure 42):

Figure 42: Tilting mechanisms alternatives based on the suspension and body independence

A: Separate body from suspension

In this case the front suspension does not lean, it maintains as if the vehicle did not have a tilting degree of freedom and the wheels maintain vertical as well. Therefore, the suspension keeps its design objective: to absorb the vibration and the bumps reactions coming from the irregular profile of the ground. The current design of the suspension could be used for this tilting PEV.

On the other side, the junction part between the two parts supposes a high risk point, since all the forces in the vehicle will be concentrated in that point. In addition, it would require to re-design the steering mechanism, since there is a mismatch between the two parts.

- √ Suspension does not lean, wheels maintain vertical
- √ Suspension used for vibration absorption
- √ No changes in suspension design

- × Robust design required, risk point
- × Concentration of forces in the joint
- × Steering re-design

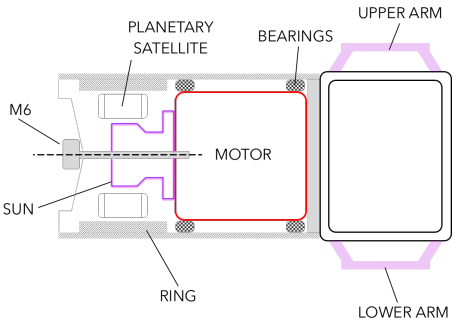


Figure 43: Design proposal in the joint with option A

B: Body moves with suspension

When the vehicle leans with the suspension, then the wheels also tilt (the camber changes) and there is no risk point since all the frame parts are joint together. Apart from small changes, the steering system can be maintained, which is a advantage. However, this alternative also gives rise to other issues. Leaning the wheels can imply an instability in the vehicle, since some tires are not designed to be tilted an elevate angle. There are also a great deal of actuation strategies, which can be difficult to choose from.

Nevertheless, the main drawback of this design is that the suspension is not anymore independent from the leaning on the vehicle. This implies that the suspension shock absorbers need to change their position and that the motor actuation cannot conflict with the motion of the suspension due to vibrations or bumps. This issue is really challenging, and in this project we were not able to design a front suspension that separated the suspension motion from the tilting motion. We will sacrifice the absence of the suspension for the sake of reaching to a new concept of tilted tricycle.

- √ Suspension and wheels do lean
- √ Robust design: no risk point
- √ Keep steering design

- × Suspension not independent from leaning
- × Multiple actuation strategies
- × Stability of the wheels

To understand the different tilting suspensions, the program Autodesk ForceEffect software was used as a first approach. The suspension was modelled in a 2D plane, with only 1 degree of freedom (which accounts for the leaning of the suspension).

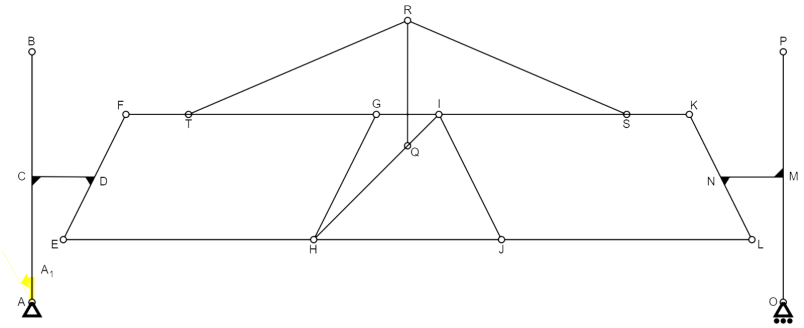


Figure 44: Front suspension model in 2D

The main parts and the links between them are, from left to right:

- *Wheel (Left)*: Co-linear bars $\overline{AC}$ and $\overline{BC}$. Point A is in contact with the ground by an articulation. Distance $\overline{AC}$ represents the radius of the front wheel.
- *Kingpin (Left)*: Co-linear bars $\overline{DE}$ and $\overline{DF}$. It is inclined by the kingpin angle.
- *Hub (Left)*: $\overline{CD}$ bar, welded to the left wheel and the kingpin.
- *Upper/Lower Suspension Arms*: bar $\overline{FG}/\overline{EH}$ articulated in both sides.
- *Body*: it represents the frame of the vehicle, where the driver will be sitting. Four-sided solid body $GHIJ$.
- *Rotation Point*: Q is the point from which the motor will actuate.
- *Tilting Mechanism*: Bars $\overline{QR}$ and $\overline{RT}/\overline{RS}$.
- *Wheel (Right)*: Co-linear bars $\overline{QO}$ and $\overline{OR}$. Point Q is in contact with the ground by a slider.

With this basic model in mind, the four studied alternatives can be summarized as follows. A more detailed analysis of the front suspension will be explained in the next chapter as well.

- **B.1 Rotating point on top of body + linkage (shock absorbers) to upper arms**

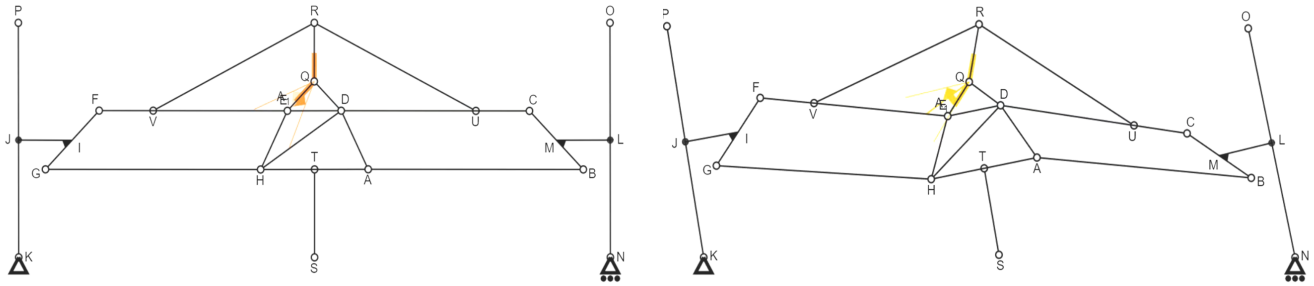


Figure 45: B_1 option

The tilting mechanism consists of three elements. The arm $\overline{QR}$ is actuated by the motor, and the two remaining bars ($\overline{RU}$ and $\overline{RV}$) are connected to the upper arms. This is the bar system used in the miniPEV. It is a very simple mechanism, but in terms of geometry, requires an angle of **45 degrees** between the $\overline{QR}$ and $\overline{RU}$ bars in order to get maximum torque on the suspension bars. The rotation point Q is constrained to be on top of the body, and considering the length of the $\overline{QR}$ bar, the space requirement is high.

– **B.2 Rotating point below body + linkage (shock absorbers) to lower arms**

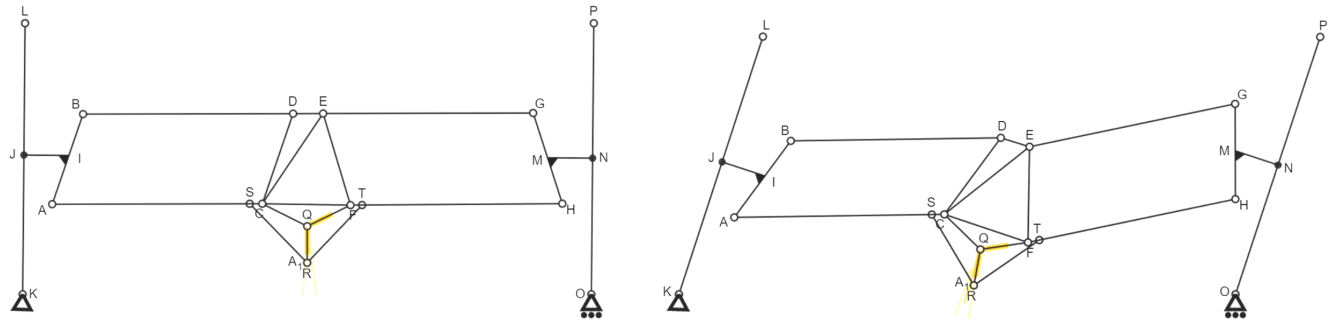


Figure 46: B_2 option

This strategy is very similar to the B.1, but the motor is located below the body. This can create problems with the motor touching the ground in some circumstances, something that should be avoided. If compared to the previous option, this alternative is not recommended.

– **B.3 Rotating point inside body + horizontal shock absorbers to lower arms**

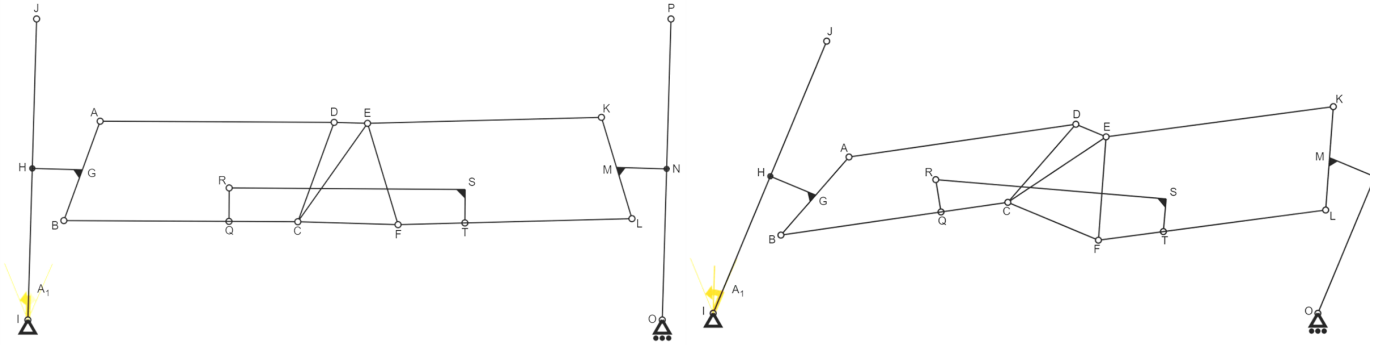


Figure 47: B_3 option

Using a rotary motor and a rack-pinion system, the rotational motion can be transformed into a linear (horizontal) motion, thus changing the distance between the lower suspension arms and tilting the vehicle.

– **B.4 Rotating point inside body + gearbox to lower arm**

By means of a gearbox, the torque coming from the motor is increased, as the suspension arms are actuated art the same time. The simplicity and freedom to choose the reduction ratio were the main reasons why this option was designed, fabricated and assembled in the full scale PEV.

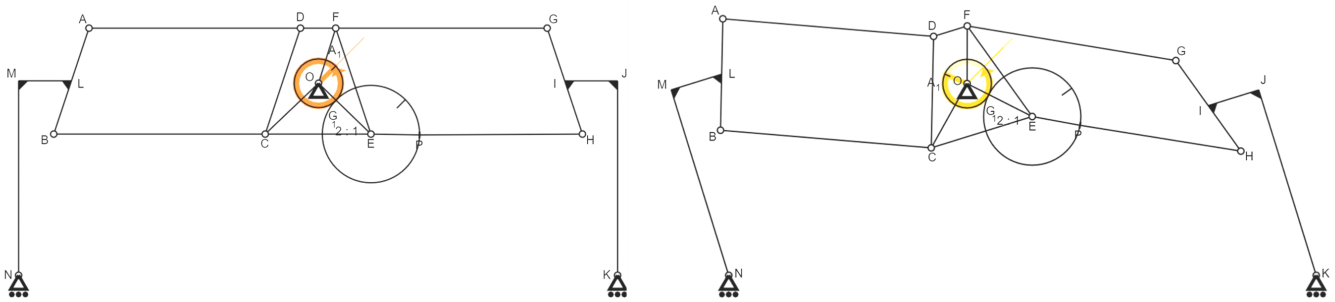


Figure 48: B_4 option

As has already been stated, the miniPEV will have a tilting mechanism similar to the option B.1 whereas the full scale PEV will be designed following the B.4 design.

First Sketches

At first, some drawings were made to illustrate the model in three dimensions. In Figure 49, the tilt actuator and the arms with the shock absorbers activate the tilting on the trike. The suspension is constituted by a 4-bar mechanism; the simplest bar system that allows a tilting motion.

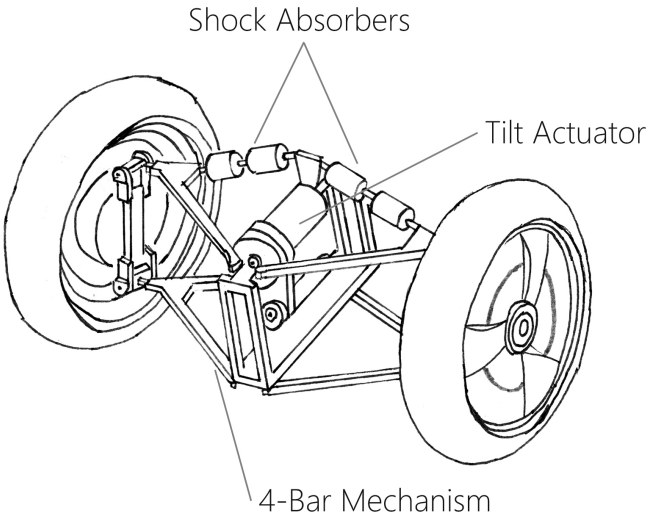


Figure 49: Sketch of tilting mechanism

The front and rear views of the suspension (Figure 50) clarify the 4-bar mechanism mentioned previously. When the vehicle runs into a bump, the shock absorbers in the affected side compress, thus minimizing the vibrations that reach to the driver. In addition, the suspension allows the tilting of the vehicle an angle θ, which will be limited by the geometry, but should not exceed 30 degrees.

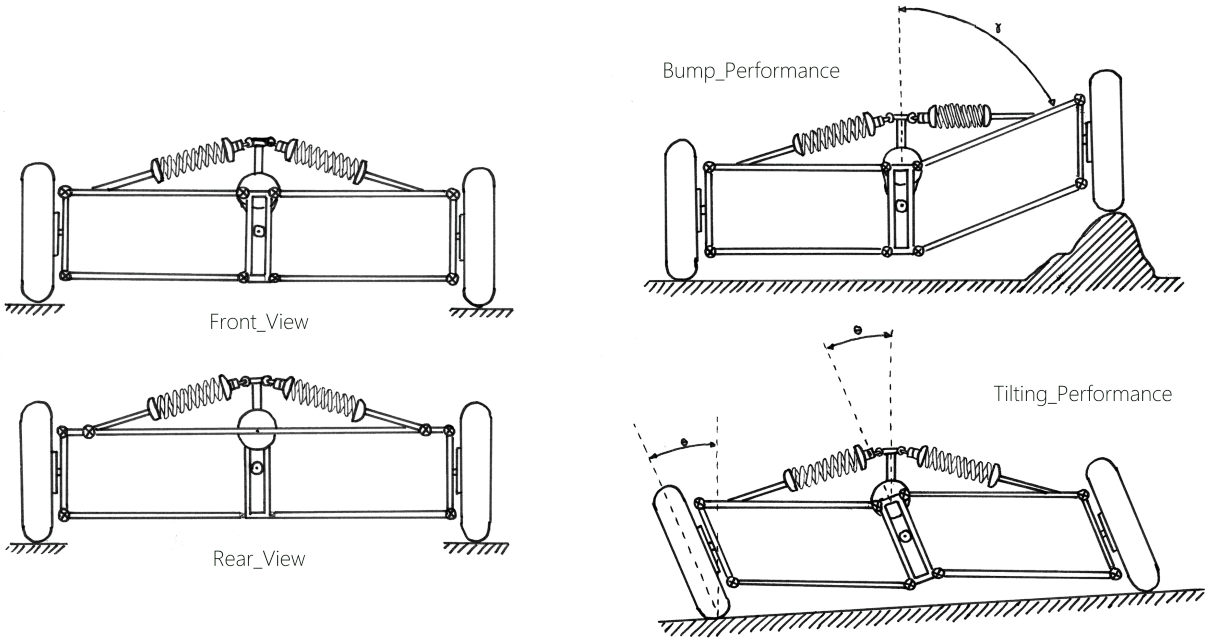


Figure 50: Sketches with response to bump and tilting

In Figure 51 more detailed drawings of the suspension have been included.

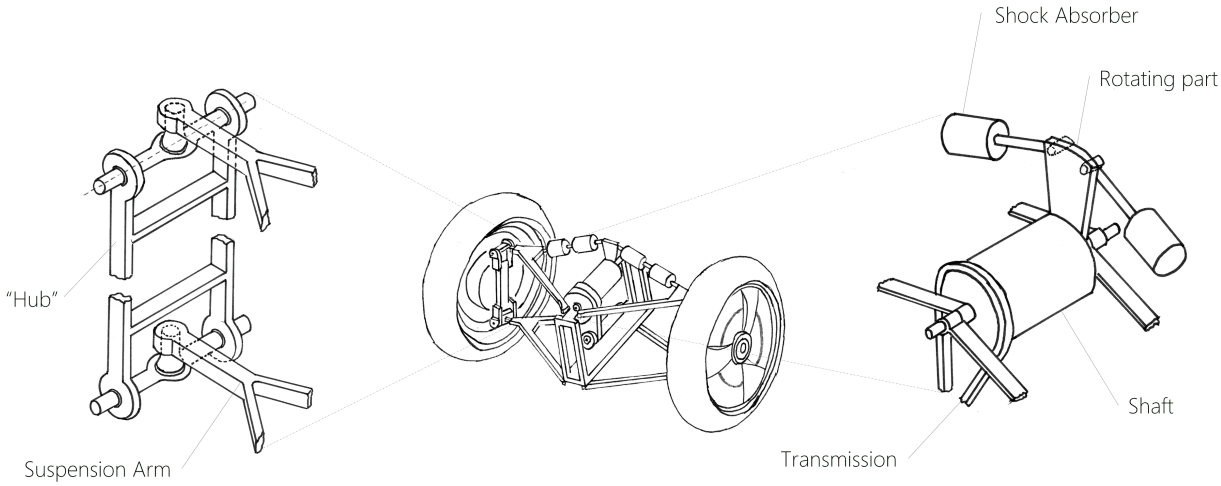


Figure 51: Detailed sketches of the tilting suspension

Components

In this section the components of the miniPEV and their main characteristics are reviewed.

Figure 52: miniPEV explosion render

– Wood Frame and Wheels

  Taking as example other previous wood models from the group, the vehicle was designed in Solidworks. The body is made of 3mm plywood sheets, and the rear wheel is slightly bigger than the front ones. Using the laser cutting machine, the body, the wheels, and the rest of the parts were fabricated quite fast. The fact that the model was made out of plywood brought along a lot of issues during the motion of the vehicle. This type of prototypes are not usually designed to move, but to remain static instead. In any case, there was a lot of effort in reducing the friction between the moving parts (wheels and steering system).

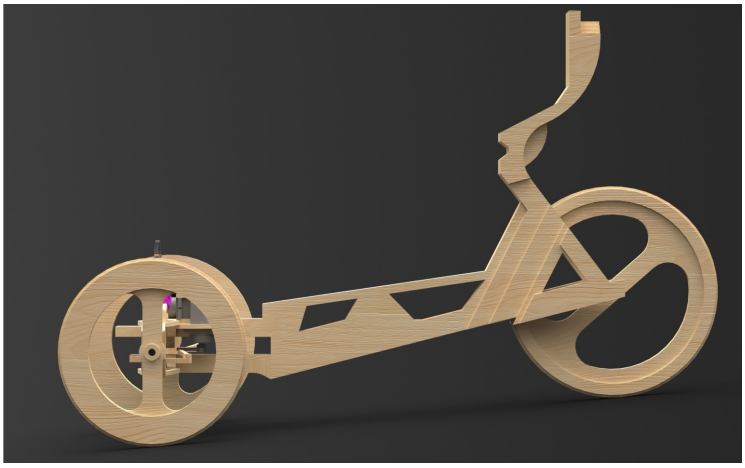


Figure 53: Lateral view of the render

– Rear DC Motor

The rear motor is a Pololu DC motor of 12V and a gearbox integrated. The gearbox fits perfectly in this application, providing a high output torque to be able to move the vehicle. The torque increase is about 50:1, and the motor's small size allows it to hide in the frame.

Figure 54: Rear DC Motor, Pololu 50:1 Micro Metal Gearmotor

Controlling a DC motor with Arduino requires a transistor and a diode. The small DC motor is likely to use more power than an Arduino digital output can handle directly (50mA). Connecting the motor straight to an Arduino pin would damage the Arduino. Tha is why a small transistor like the PN2222 needs to be used as a switch, that uses just a little current from the Arduino digital output to control the much bigger current of the motor.

There is also a diode connected across the connections of the motor. Since diodes only allow electricity to flow in one direction, when to motor turns off the diode protect the Arduino and the transistor from a negative spike of voltage. The diode protects against this, by shorting out any such reverse current from the motor.

Figure 55: Servo Motor MG90

– Steering and Tilting

The steering servo motor is located vertically in the left side of the vehicle, and has two bars linked to the hubs of the front wheels. The tilting motor is laying on top of the body, horizontally, and has a glass arm attached to the output shaft. This glass piece is connected to the shock absorbers that are attached to the upper suspension arms.

Figure 56: Miniature shock absorbers MA 35 to MA 900

The tilting servo motor is slightly more powerful than the steering motor, since the force requirement for the tilting motion is higher than for turning the wheels.

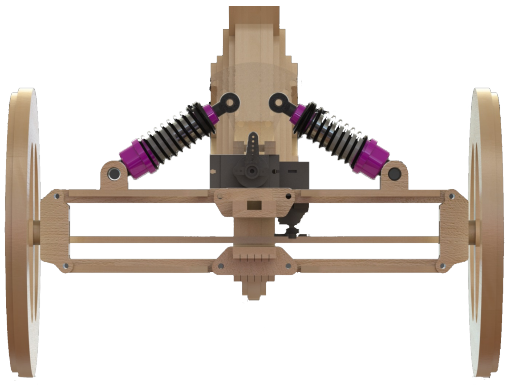


Figure 57: Front view of the render

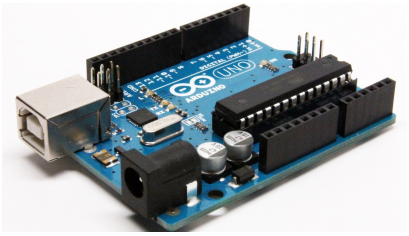
– Electronics

The **Arduino UNO** is the board responsible for receiving the inputs from the sensors and controlling the output to the three motors. The board and all the components are powered by a 12V DC battery (**Energizer XP8000**). The election of the 12V is due to the rear motor input voltage. The rest of the components (servo motors, bluetooth module and IMU) only require 5V, voltage that the Arduino board can provide from one of its pins. All these electronic components are placed in a mini breadboard.

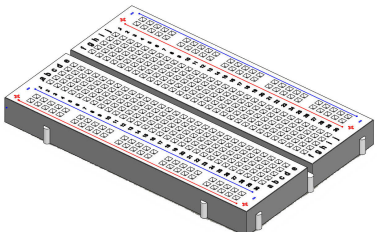
The chosen IMU both for the miniPEV and for the PEV is the **Adafruit BNO055 Absolute Orientation Sensor**, with 9DOF (accelerometer, gyroscope and magnetometer). It is connected through the I2C connection (SDA and SCL pins) to the board. The measurements from this sensor are the angular velocity along the vertical axis Z (**yaw rate** $\dot{\psi}$) and the **linear acceleration** in the forward direction a_x, which will be integrated to estimate the longitudinal speed of the vehicle.

As a matter of fact, the use of this Inertial Measurement Unit is quite easy, since the libraries to get the data have been already developed. Nevertheless, there are some steps that need to be followed in order to configure the sensor in the first place (this will be intensively explained in the next chapter).

Figure 58: Electronic Components

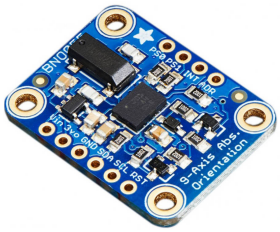


a) Arduino UNO board



b) Mini breadboard

c) Energizer XP8000 battery



d) BNO055 9DOF IMU

Finally, in order to remotely connect with the board and control the vehicle, a Bluetooth HC05 module was installed. This module was connected through serial connection to the Arduino. An Android application was the installed in the phone to activate different functions (forward, left, right...). The wiring diagram is represented as follows:

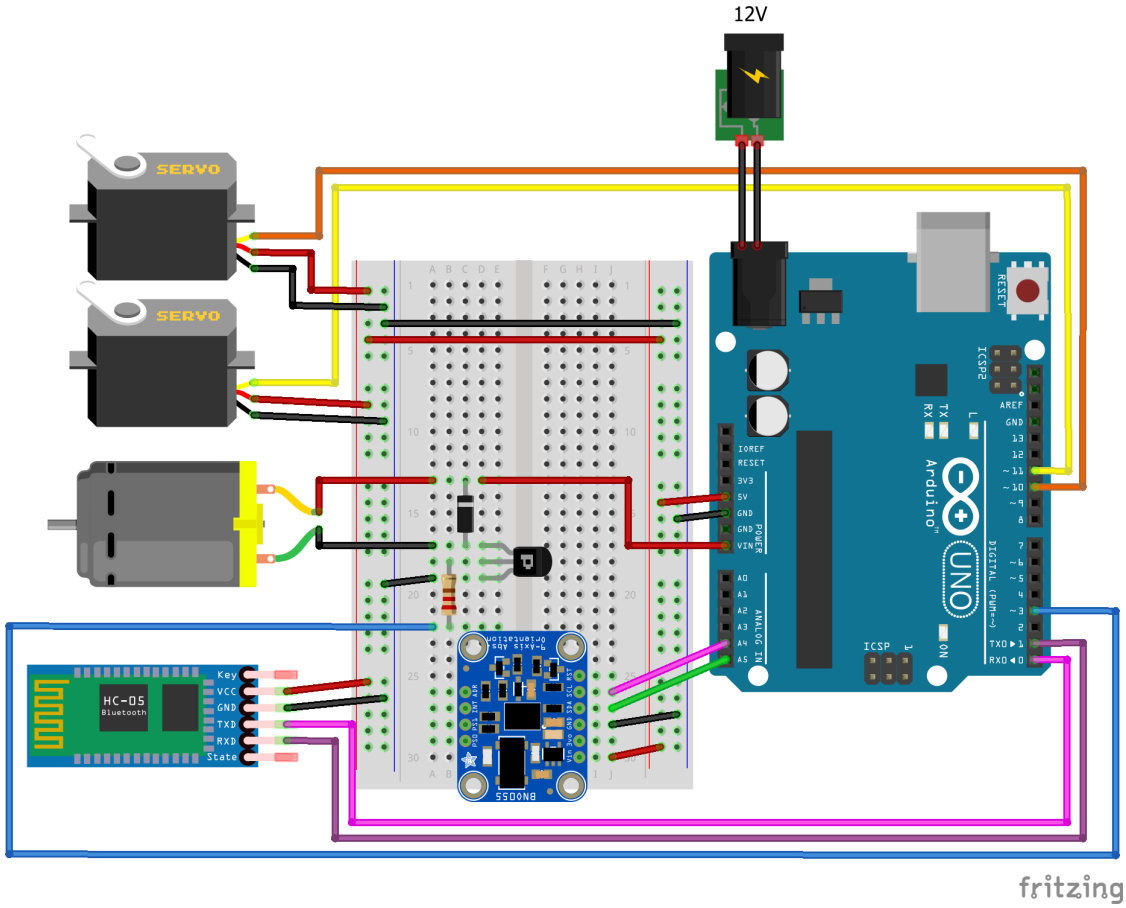


Figure 59: miniPEV Electronics Schematic

Final Model and Results

The final model is illustrated in the Figures 61 and 62. The tilting servo motor is controlled to satisfy in every moment that $\theta = \theta_{ref}$, with

$$\theta_{ref} = \frac{V\dot{\psi}}{g}$$

This expression of the roll angle reference was introduced in the previous chapter and is based on a very simple model that combines the bicycle and the inverted pendulum dynamics (Figure 60).

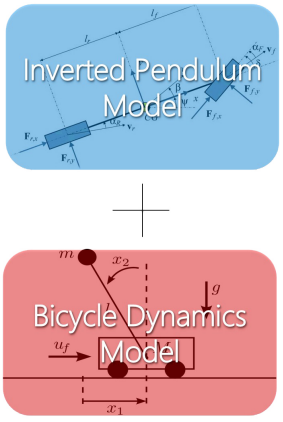


Figure 60: miniPEV simple model to obtain the θ_{ref}

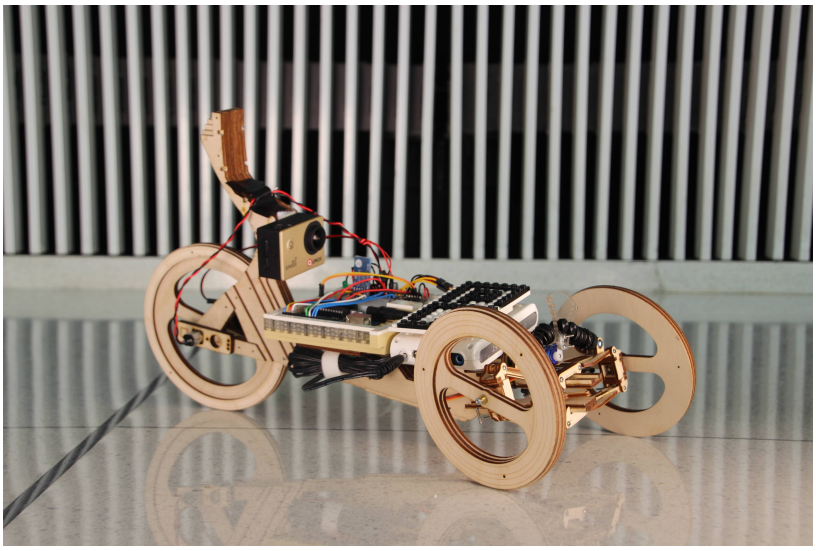


Figure 61: Lateral view of the miniPEV

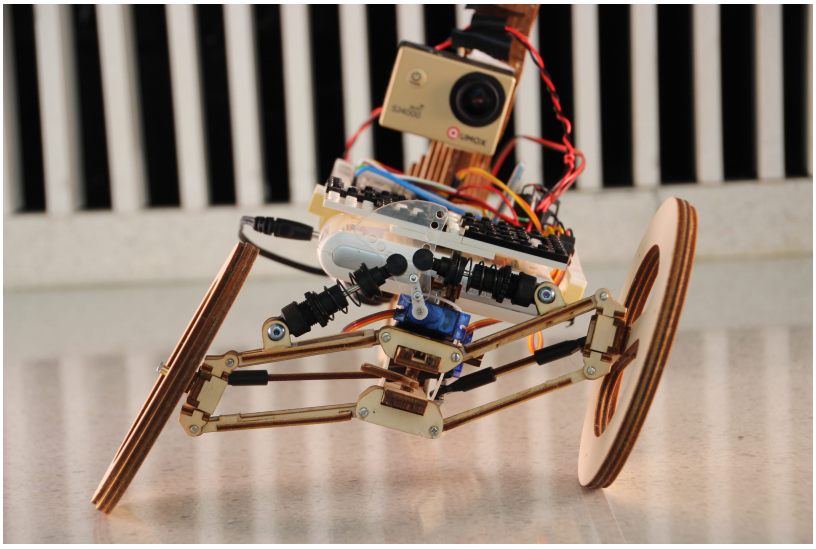


Figure 62: Front view of the miniPEV

The value for the angular velocity (yaw rate) $\dot{\psi}$ is obtained from the gyroscope sensor measurement. However, there is no direct information about the longitudinal velocity of the vehicle. That is why the linear acceleration in the forward direction a_x is integrated, thus estimating the velocity V.

A strategy in two steps was carried out to filter and integrate the linear acceleration signal. First, a Kalman filter was used and, in second place, a more manual filtering was implemented. In order to validate the integration algorithms, some live tests were made. These test samples were obtained from the Media Lab elevator, going up and down several times.

Kalman Filter

A Kalman filter is an optimal estimation algorithm used to estimate states of a system from indirect and uncertain measurements. The most simple Kalman filter has some variables: x for the filtered value, q for the process noise, r for the sensor noise, p for the estimated error and k for the Kalman Gain.

The filter is applied with each measurement and initialized with the process noise q, the sensor noise r, the initial estimated error p and the initial value $x = 0$. The initial value for p is not very important since it is adjusted during the process. It must be just high enough to narrow down. The filter can be summed up in two simple steps:

- Predict

  The current state x_k is estimated from the previous state x_{k-1} and the estimated error p increases by the inherent process noise q.

  $$x_k = x_{k-1}, \quad p = p + q$$

- Measurement Update

  The kalman gain k balances the relevance of the current observation z_k and the previous state $\hat{x}_{k-1}$. It is the ratio of the estimated error p and the noise coming from the sensor r.

  $$k = \frac{p}{p+r}, \quad \hat{x}_k = \hat{x}_{k-1} + k(z_k - \hat{x}_{k-1})$$

  When the gain is 0, the current observation has no effect on the updated value, whereas if its value is 1 the previous state does not have an influence.

After filtering the acceleration a_x with $x_0 = 0 \quad p = 1 \quad q = 0.01 \quad r = 0.15$, the velocity is integrated with the **trapezoidal rule of numerical integration**:

$$\hat{v}_k = \hat{v}_{k-1} + (a_k + a_{k-1})\frac{\Delta t}{2}$$

The issue with this strategy is the shift of the velocity origin due to the noise in the acceleration and the error in the time interval. These errors accumulate during the ups and downs in the elevator, and incorrectly gives non-zero values of the velocity.

There is an extended Kalman filter that accounts for this shift issue, but is designed to correct the possible shifts in the sensors signals, that is, to fix any possible deviation in the signal coming from the IMU. In this case, we are trying to deal with a numerical integration, not a shift in the signal from the IMU. Therefore, this path is not applicable to this case.

The Kalman filter is perfect for filtering the noise from the accelerometer, but it is not enough to give a good estimation of the velocity when integrating the filtered acceleration.

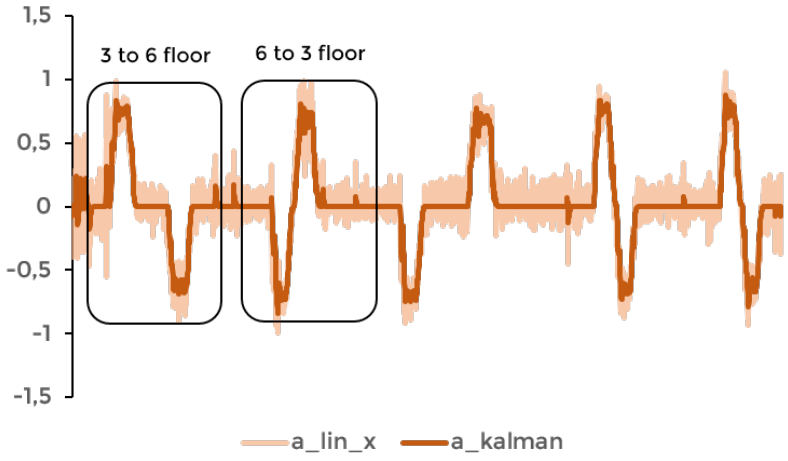


Figure 63: Kalman filtering to elevator test

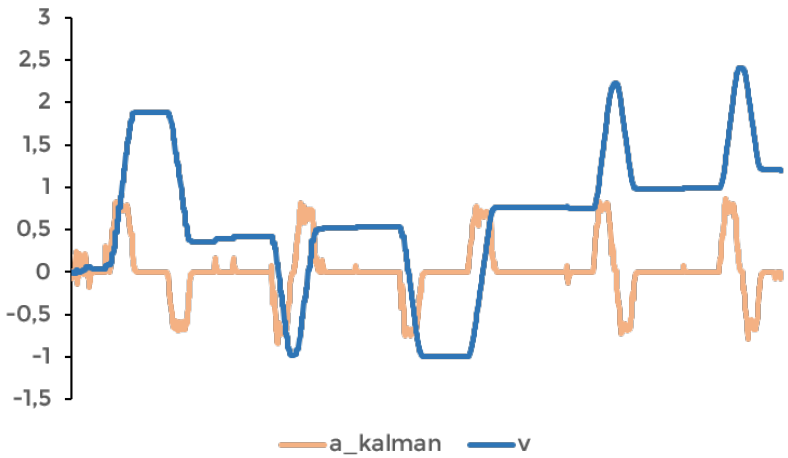


Figure 64: Elevator velocity estimation with trapezoidal rule

Integration Shift

In order to fix the accumulative error in the integration of the acceleration, a very simple algorithm was implemented. This algorithm just corrected the shift by estimating the slope (m) of the error during an early sample.

- Trapezoidal

$$\hat{v}_k = \hat{v}_{k-1} + (a_k + a_{k-1})\frac{\Delta t}{2}$$

- Corrected:

$$\hat{v}_k = \hat{v}_{k-1} + (a_k + a_{k-1})\frac{\Delta t}{2} - m\,\Delta t$$

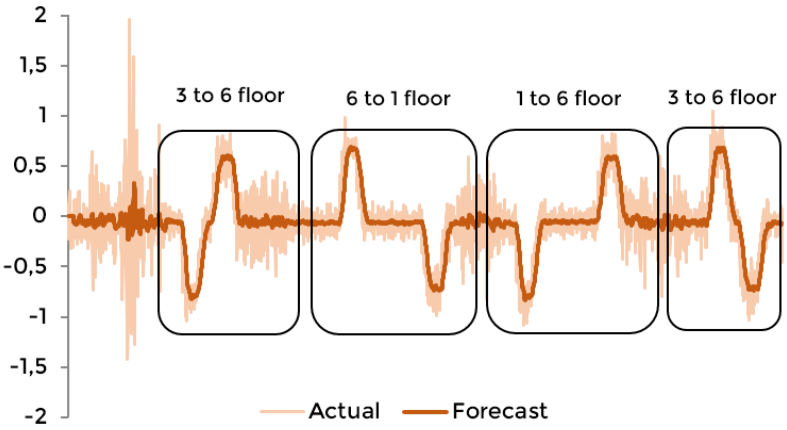


Figure 65: Sample from the elevator

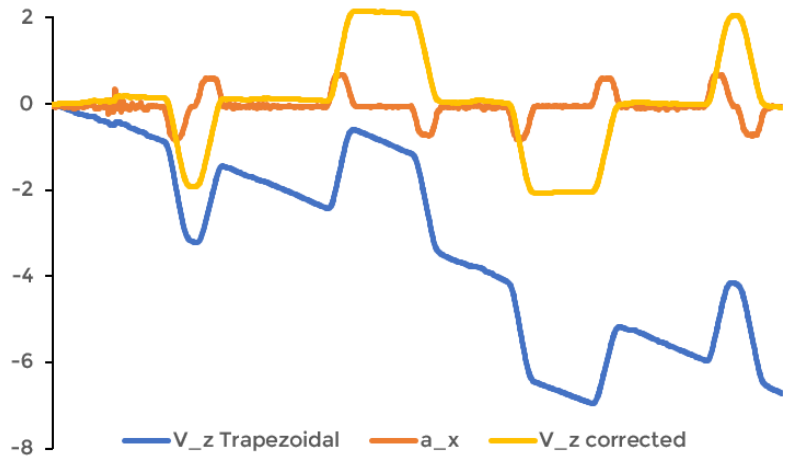


Figure 66: Acceleration, and integration with trapezoidal rule and a corrected estimation

After calibrating this algorithm, a final test was carried out in the elevator, giving good results (Figure 67).

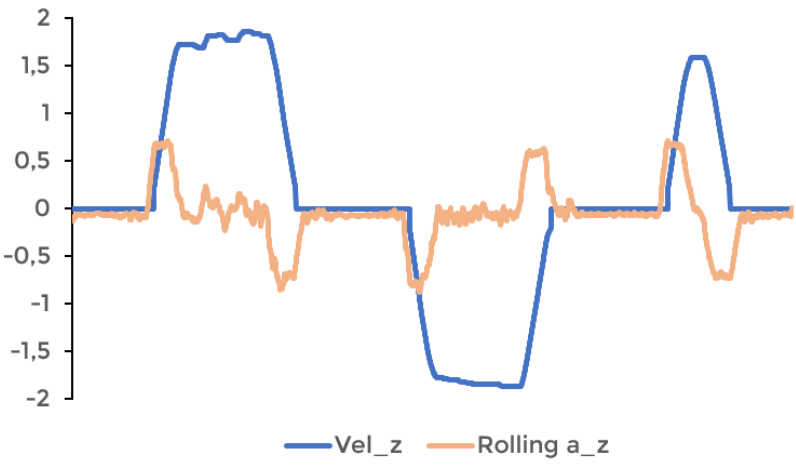


Figure 67: Final test in the elevator

Chapter Conclusions

The design and fabrication of the miniPEV supposed the first step in this project, and has been really useful for understanding the possible problems that could rise in a tilting vehicle of bigger dimensions. Sometimes looking at a prototype for five minutes is much more productive than thinking about its design and making hundreds of sketches. It also was useful to explain to the Changing Places group and other labmates the ideas that were being developed.

Figure 68: miniPEV stand at the Media Lab Members Event Fall 2016

In the technical side, there were some challenges. First, the estimation of the longitudinal speed of the vehicle required several tests and some time going up and down in the Media Lab elevator (which was kind of fun). The results of these test were useful for the integration of other variables in the full scale vehicle.

Regarding the tilting control, the delay between the input from the driver and the action from the motor was evident. This agrees with the studies mentioned previously in the literature review.

Overall, the vehicle satisfactorily performed as expected, and the tilting worked perfectly, even though the motor-actuated glass part was larger than in the initial design. In the next chapter the real scale PEV will be presented.

Figure 69: Remote control from the phone to the miniPEV via Bluetooth

5. Real Scale Prototype

Once the dynamic model and the control strategy have been studied, the design and fabrication of the tilting PEV will be explained in this chapter. The frame of this vehicle was recycled from another previous PEV version, and will be the departure point for this new prototype. In this chapter we will cover the processes to design and fabricate all the parts in the vehicle, (front suspension, tilting and steering mechanisms, rear motor, handle bar, batteries, electronic components...)

Front Suspension Design

The first step is to design the front suspension of the vehicle. The suspension arms will continue to form a four bar mechanism as were in the miniPEV. In this case, before designing any mechanical component and jumping to the CAD software, some kinematic and dynamic simulations were carried out. These simulations had several goals in mind.

1. Understand the motion of the tilting suspension depending on its geometry.
2. Maintain the wheels as vertical as possible during the leaning of the body.
3. Minimize the required torque from the tilting motor.

The simulations were developed in MATLAB, following a similar structure to the functions introduced in the book by A. Avello[62]. The analysis starts by defining a model of elements that are linked together by different types of joints. This model is then translated into a set of equations Φ that define the constraints of the system. Whichever the motion of the mechanism, it must obey these equations.

[62] Alejo Avello Iturriagagoitia. *Teoría de máquinas Segunda Edición*. Escuela Superior de Ingenieros Industriales - Universidad de Navarra, 2014. Available at http://hdl.handle.net/10171/34797

The kinematic simulations require to solve three problems (position, velocity and acceleration) in that order. Then the dynamic simulation makes use of the results from the kinematic analysis, giving an estimation of the forces and reactions in the system.

Model Schematic

The first step to a mechanical analysis is to generate a model of elements that represent the system. In this case, the front suspension is abstracted into a 2D model of bars.

The process of designing the model is very simple. The user introduces the length of the bars, then the coordinates of all points are calculated from a fixed frame reference and based on those bar lengths. Finally, the estimated mass of each bar is introduced, and all possible external forces are applied as well. For example, the weight of the driver is simulated as a 80kg weight located in a point near to the center of gravity of the vehicle.

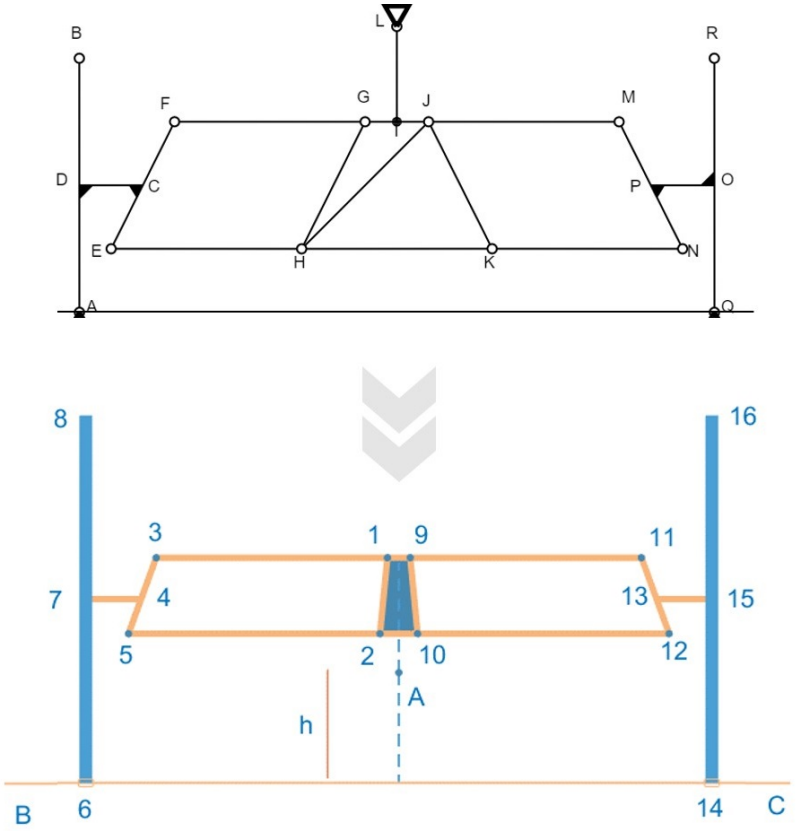


Figure 70: Simulated Model

Based on the Autodesk ForceEffect model designed previously, the suspension model was implemented in MATLAB. It has 20 points – 3 of them fixed, A, B, C – that define the bars. There are only three types of links in the model: articulations, weld and slider.

The constraint equations that define this model can be summarized in these types (the examples have been taken from the model in Figure 70):

x_i is the x coordinate of the point i

y_i is the y coordinate of the point i

$L_{a\,b}$ is the length of the bar *a*-*b*

- Bar $(x_1 - x_3)^2 + (y_1 - y_3)^2 - L_{13}^2 = 0$
- Triangle
$$\begin{aligned}(x_1 - x_2)^2 + (y_1 - y_2)^2 - L_{12}^2 = 0\\ (x_1 - x_9)^2 + (y_1 - y_9)^2 - L_{19}^2 = 0\\ (x_2 - x_9)^2 + (y_2 - y_9)^2 - L_{29}^2 = 0\end{aligned}$$
- Welded
$$\begin{aligned}(x_6 - x_7)(x_4 - x_7) + (y_6 - y_7)(y_4 - y_7) - L_{67}L_{47}\cos(90) = 0\\ (x_6 - x_7)(y_4 - y_7) - (y_6 - y_7)(x_4 - x_7) - L_{67}L_{47}\sin(90) = 0\end{aligned}$$
- Slider $(x_6 - x_B)(y_B - y_C) - (y_6 - y_B)(x_B - x_C) = 0$

Each of these constraint equations is denoted as $\Phi(\vec{g}, t)$, since depend in the vector of the coordinates $\vec{g}$ and the time t.

In Figure 70 only 16 moving and 3 fixed points have been represented. The model will be completed with the introduction of the tilting actuation and the driver model. In total there will be 20 moving points and 3 relative coordinates (angles), which make a total of **43 unknown coordinates**.

$$\vec{g} = \begin{bmatrix} x_1 & y_1 & x_2 & ... & x_{20} & y_{20} & \theta & \gamma & \phi \end{bmatrix}_{43x1}$$

$$\Phi(\vec{g}, t)_{42x1}$$

$$\Phi_g(\vec{g}, t)_{42x43} \quad with \quad \Phi_g(\vec{g}, t)_{i,j} = \frac{\partial \Phi_i}{\partial x_j}$$

Note that the size of the coordinates vector (unknowns) is bigger than the number of equations in the system (42 $\rightarrow$ 43). This is due to the fact that one of the coordinates will be degree of freedom, and therefore a known value.

Due to the geometry of the model, it is impossible to put one fixed articulation in each wheel's point of contact with the ground. Instead, one of these two links has to be modified into an articulated slider. Another alternative was the implemented design, with a fixed rotation point and one articulated slider on each wheel. With this decision, the model behavior is completely identical –symmetrical– on both sides.

From now on we will refer as **inner wheel** to the wheel closer to the center of rotation of the trajectory and as **outer** to the other one. This notation will be alternated with left and right wheels for inner and outer.

Kinematic Simulations

The goal of the kinematic simulations is to find the best geometry for stability during the tilting motion, that is, to design the front suspension so that the inner wheel should maintain as vertical as possible, so the outer wheel.

The degree of freedom z of the model is the rotation angle ϕ that the body forms with the vertical axis. The vector of coordinates $\vec{g}$ is formed by the x and y coordinates of the 20 points. Other extra variables, angles and distances of interest are also included in the vector $\vec{q}$. As stated previously, to study the motion of a system three problems have to be solved, in this order:

- Position Problem

  Position Problem:

  Input z; Unknown g

  Given a value for the degree of freedom $z = \phi$, the vector of coordinates is calculated solving the system of non-linear equations Φ. The Newton-Raphson method is used to solve the equations:

  $$\Phi(\vec{g} + \Delta\vec{g}, t) \approx \Phi(\vec{g}, t) + \Phi_{\vec{g}}(\vec{g}, t)\Delta\vec{g} \approx 0$$

  where the $\Phi_{\vec{g}}$ is the Jacobian of the constraint equations. Through a iterative process, the values of the coordinates are calculated:

  $$\Phi_{\vec{g}}(\vec{g}_i, t)(g_{i+1} - g_i) = -\Phi(g_i, t)$$

- Velocity Problem

  Velocity Problem:

  Input $g, \dot{z}$; Unknown $\dot{g}$

  The derivative of the constraint equations is also null:

  $$\frac{d}{dt}\Phi(\vec{g}, t) = \Phi_{\vec{g}}\,\dot{\vec{g}} + \Phi_t = 0$$

  where Φ_t is the partial derivative of the constraint equations with respect to time t. The position of the system is known for each time t, so the matrix $\Phi_{\vec{g}}$ and Φ_t are known, which gives the values of the velocity $\dot{\vec{g}}$:

  $$\Phi_{\vec{g}}\,\dot{\vec{g}} = -\Phi_t$$

- Acceleration Problem

  Acceleration Problem:

  Input $g, \dot{g}, \ddot{z}$; Unknown $\ddot{g}$

  Following the same logic, the vector of accelerations is obtained:

  $$\frac{d^2}{dt^2}\Phi(\vec{g}, t) = \Phi_{\vec{g}}\,\ddot{\vec{g}} + \dot{\Phi}_{\vec{g}}\,\dot{\vec{g}} + \dot{\Phi}_t = 0$$

  . If this equation is rearranged, the only unknown is the vector of accelerations $\ddot{\vec{g}}$:

  $$\Phi_{\vec{g}}\,\ddot{\vec{g}} = -\dot{\Phi}_{\vec{g}}\,\dot{\vec{g}} - \dot{\Phi}_t$$

Once the kinematic problem has been solved, the dynamic of the system can be studied. From the principle of virtual work:

$$\delta W_{inertia} + \delta W_{external} = 0$$

Applying this theorem in function of the vector of natural coordinates g:

$$\delta \vec{g}^T \, M \, \ddot{\vec{g}} + \delta \vec{g}^T \, Q = 0 \tag{53}$$

where M is the mass matrix and Q the vector of generalized forces.

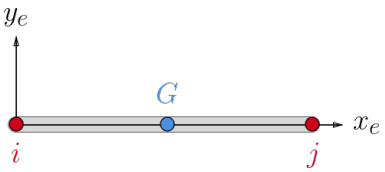


Figure 71: Element of the model

To build M, each element's M_e is calculated first, and its inserted in the corresponding cells of the M matrix.

$$M_e = \begin{bmatrix} m + a - 2b_x & 0 & b_x - a & -b_y \\ & m + a - 2b_x & b_y & b_x - a \\ & & a & 0 \\ sim. & & & a \end{bmatrix}$$

with

$$a = \frac{I_i}{L_{ij}^2} \quad b_x = \frac{m \, {}^e x_G}{L_{ij}} \quad b_y = \frac{m \, {}^e y_G}{L_{ij}} \quad m = \int_V dm$$

$$I_i = \int_V ({}^e x^2 + {}^e y^2) dm \quad {}^e x_G = \frac{1}{M} \int_V {}^e x \, dm \quad {}^e y_G = \frac{1}{M} \int_V {}^e y \, dm$$

The matrix Q is build in the same way, but accounts for the external forces in the system, for example, the torque from a motor. For any force F in the model:

$$Q_F = \frac{1}{L_{ij}} \begin{bmatrix} L_{ij} - {}^e x & {}^e y \\ {}^e y & L_{ij} - {}^e x \\ {}^e x & {}^e y \\ -{}^e y & {}^e x \end{bmatrix} \begin{bmatrix} F_x \\ F_y \end{bmatrix}$$

The matrix M and Q are therefore known. The values of the coordinates vector $\vec{g}$ can be expressed as a function of the degrees of freedom z as $\vec{g} = f(z)$ where

$$\dot{\vec{g}} = \frac{\partial f}{\partial z} \dot{z} = R \, \dot{z} \qquad \ddot{\vec{g}} = R \, \ddot{z} + \dot{R} \, \dot{z}$$

Returning to the equation (53):

$$\begin{aligned} \delta \vec{g}^T (M \ddot{\vec{g}} - Q) &= 0 \\ \delta z^T R^T (M \ddot{\vec{g}} - Q) &= 0 \\ R^T (M \ddot{\vec{g}} - Q) &= 0 \\ R^T M \ddot{\vec{g}} &= R^T Q \\ R^T M R \ddot{z} &= R^T Q - R^T M \dot{R} \dot{z} \end{aligned}$$

Thus the acceleration of the degrees of freedom is obtained when mass and external forces are applied.

Torque Calculation

It must be pointed out that in this thesis, the only variable of interest is the torque requirement from the motor. In order to get that variable a different perspective is necessary.

Once again from equation (53), the Lagrange multipliers (λ) are introduced. There is one λ per constraint equation in the model, and depending on their equation their meaning changes, but overall are related to the reactions between elements.

$$M\ddot{\vec{g}} - Q + \Phi_g^T \lambda = 0 \tag{54}$$

For getting the value of the motor torque, a known tilting angle trajectory $\phi_c(t)$ has to be imposed:

$$\Phi = \phi(t) - \phi_c(t) \tag{55}$$

For the sake of the programmer, once the kinematics have been solved, it is quite disturbing to change the system of equations again, introducing a new time depending equation. If only the motor torque is needed, a particular method to get it can be developed.

First, let us consider that the tilting angle ϕ is part of the coordinates vector $\vec{g}$ as:

$$\vec{g} = \begin{bmatrix} x_1 & y_1 & x_2 & ... & x_{20} & y_{20} & \theta & \gamma & \phi \end{bmatrix}$$

It was already verified that:

$$M\ddot{\vec{g}} - Q + \Phi_g^T \lambda = 0$$
$$\lambda = \Phi_g^H (M\ddot{\vec{g}} - Q)$$

Inserting the equation (55) does not change the terms $M\ddot{\vec{g}} - Q$ in equation (54). It only expands the Jacobian Φ_g with a row of zeros and a one:

$$\begin{bmatrix} \Phi_g \end{bmatrix} \quad \rightarrow \quad \left[\begin{array}{ccccc} & & \Phi_g & & \\ \hline 0 & 0 & ... & 0 & 1 \end{array}\right]$$

Returning to equation (54), the lagrangian terms becomes:

$$\begin{bmatrix} \Phi_g^T \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ ... \\ \lambda_{42} \end{bmatrix} \quad \rightarrow \quad \left[\begin{array}{c|c} & 0 \\ & 0 \\ \Phi_g^T(1:42,:) & ... \\ & 0 \\ \hline \Phi_g^T(43,:) & 1 \end{array}\right] \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ ... \\ \lambda_{42} \\ \lambda_{43} \end{bmatrix}$$

Calculating the product of this new lagrangian term and considering that the mass and external forces terms do not change:

$$M\ddot{\vec{g}} - Q + \Phi_g(1:42,:)^T \lambda = 0$$
$$\Phi_g^T(43,:) \begin{bmatrix} \lambda_1 & \lambda_2 & ... & \lambda_{42} \end{bmatrix}^T + \lambda_{43} = 0$$
$$M_t = \lambda_{43} = -\Phi_g^T(43,:) \begin{bmatrix} \lambda_1 & \lambda_2 & ... & \lambda_{42} \end{bmatrix}^T$$

Model

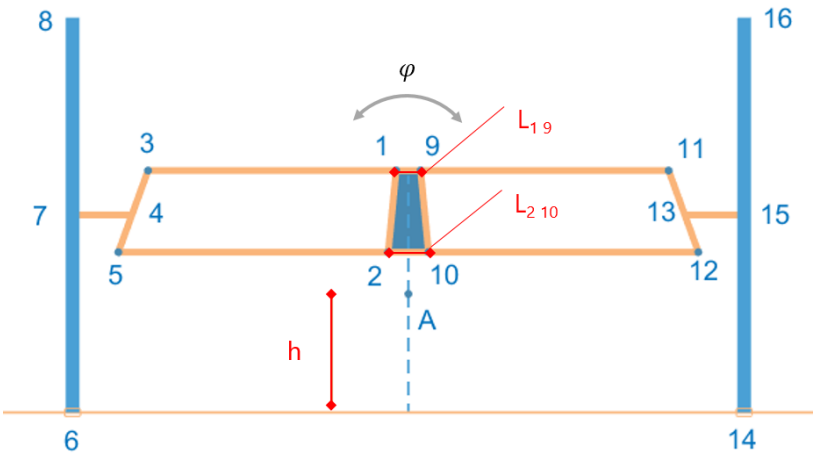


Figure 72: Model

The model has a pre-established constraints, due to the selection of some components. That is the reason why there are some fixed dimensions:

- Wheel Radius
- Vehicle Track
- Kingpin angle
- Hub width and height
- Wishbone arms horizontal in steady state

The **degree of freedom** is the body inclination ϕ, based on the rotation point A, where the motor should be placed. For the kinematic simulations it will be established that: $\phi = (0 - 30)\,°$; $\dot{\phi} = 1 rad/s$; $\ddot{\phi} = 0 rad/s^2$

Kinematics

We will study the motion of the system with varying parameters. The only parameters left to determine are the **position of the motor** –height $h \in (0 - 600) mm$– and the **width of the body in the suspension** –L_{19}, $L_{210} \in (0 - 300) mm$–, which will initially cover those ranges.

The model is fixed to some dimensions, and its **tilted to the maximum possible angle**. The inclination of the body and the wheels is recorded and saved. Then both the left and right wheels angle is compared with the body angle. This data is the **fitted into a linear regression** and its slope and quadratic error are saved for later analysis.

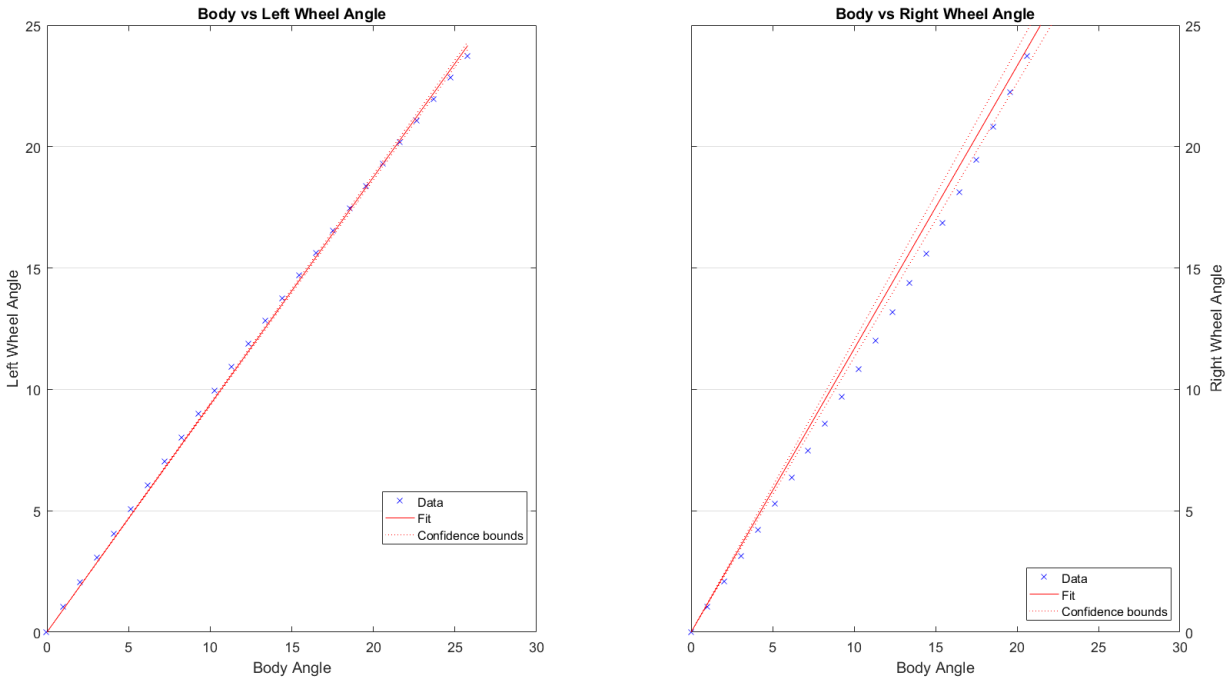


Figure 73: Left (inner) and right (outer) wheels angle versus the body angle during a tilting motion

After simulating the mechanism with different geometric constraints (54900 combinations), some conditions were applied to find the most suitable geometry:

1. Ratio between the inner wheel inclination angle and the body leaning $\phi_{inner\,to\,body} < 0.96$
2. The body width should have enough space for connecting the suspension arms L_{19}, $L_{210} > 30mm$
3. The body should lean a minimum angle ($\phi_{max} > 25°$)
4. The rotation point should be located near to the body, in order to avoid the motor scraping against the ground or being too high.

Applying these constraints a set of possibilities is obtained:

Table 5: Geometries remaining after applying the selection constraints

h	L_{19}	L_{210}	ϕ_{max}	$\phi_{left\,to\,body}$	$RSE_{left\,to\,body}$	$\phi_{right\,to\,body}$	$RSE_{right\,to\,body}$
260	31	61	0.4500	0.9530	0.0020	1.1560	0.0020
260	61	101	0.4500	0.9592	0.0017	1.1397	0.0017
250	31	61	0.4500	0.9534	0.0020	1.1598	0.0020
250	**61**	**101**	**0.4500**	**0.9596**	**0.0017**	**1.1432**	**0.0017**
240	31	61	0.4500	0.9539	0.0019	1.1432	0.0019
240	61	101	0.4500	0.9600	0.0017	1.1643	0.0017
230	31	61	0.4499	0.9543	0.0019	1.1730	0.0019

The selected geometry has been emphasized in the table 5 and represented in Figure 74. The motor should be located inside the body, at 230260 mm from the ground. The selection of L_{19} is clearly affected by the second constraint, it usually selects the lower bound (31 mm). Regarding the body geometry, the length L_{19} should be around 60 mm, while the length L_{210} should be around 100 mm. These values could be recalculated in case that the 60 mm separation between pins would not be enough.

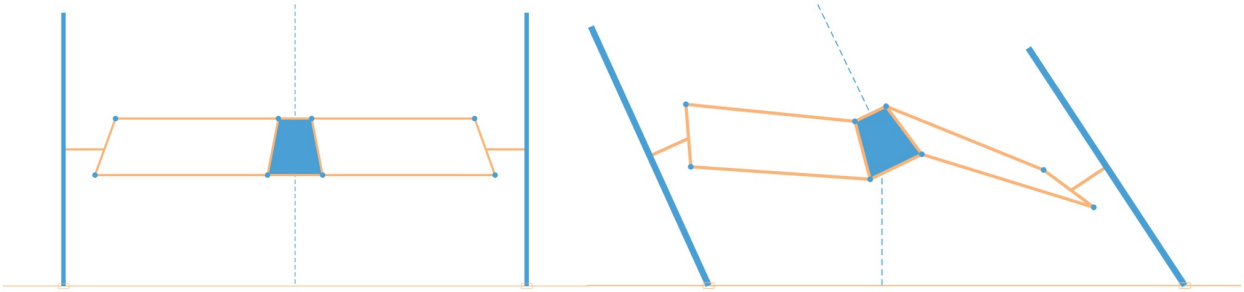


Figure 74: Selected geometry

Other interesting conclusions can be extracted from the kinematic analysis. First, let us look at the value of ϕ_{max} (Figure 75) for all the possible combinations.

We can notice that increasing the rotating point reduces the cases of high leaning angles. The lower the point A, the higher the leaning angle. In addition, ff the position of the rotating point is maintained constant ($h = constant$), that is, independently of the rotating point height, there exist a line where the dimensions L_{19} and L_{210} give the highest possible leaning angle. This line gives a body with a particular lateral edge.

Indeed, the $1\bar{2}$ edge would form 55° with the horizontal in that optimum case. But reality is that the selection has to be based in a balance, going to the optimum in one dimension will mean worse values in the other dimensions.

In Figures 76 and 77 the values of ϕ_{inner} and ϕ_{outer} tell that independently from the rotating point height, the regions where there is a low inner wheel to body angle ratio are the same where there is a high outer wheel to body angle ratio. The selected point fall into a balanced zone between these two regions.

Finally, in Figure 78 the relation between the inner-wheel-to-body ratio and the outer -wheel-to-body ratio is represented. Out of curiosity, it is impossible to find a geometry in which simultaneously the inner and the outer wheel angle are lower than the body angle. When a ratio is below 1, the other ratio is always above 1, and vice versa.

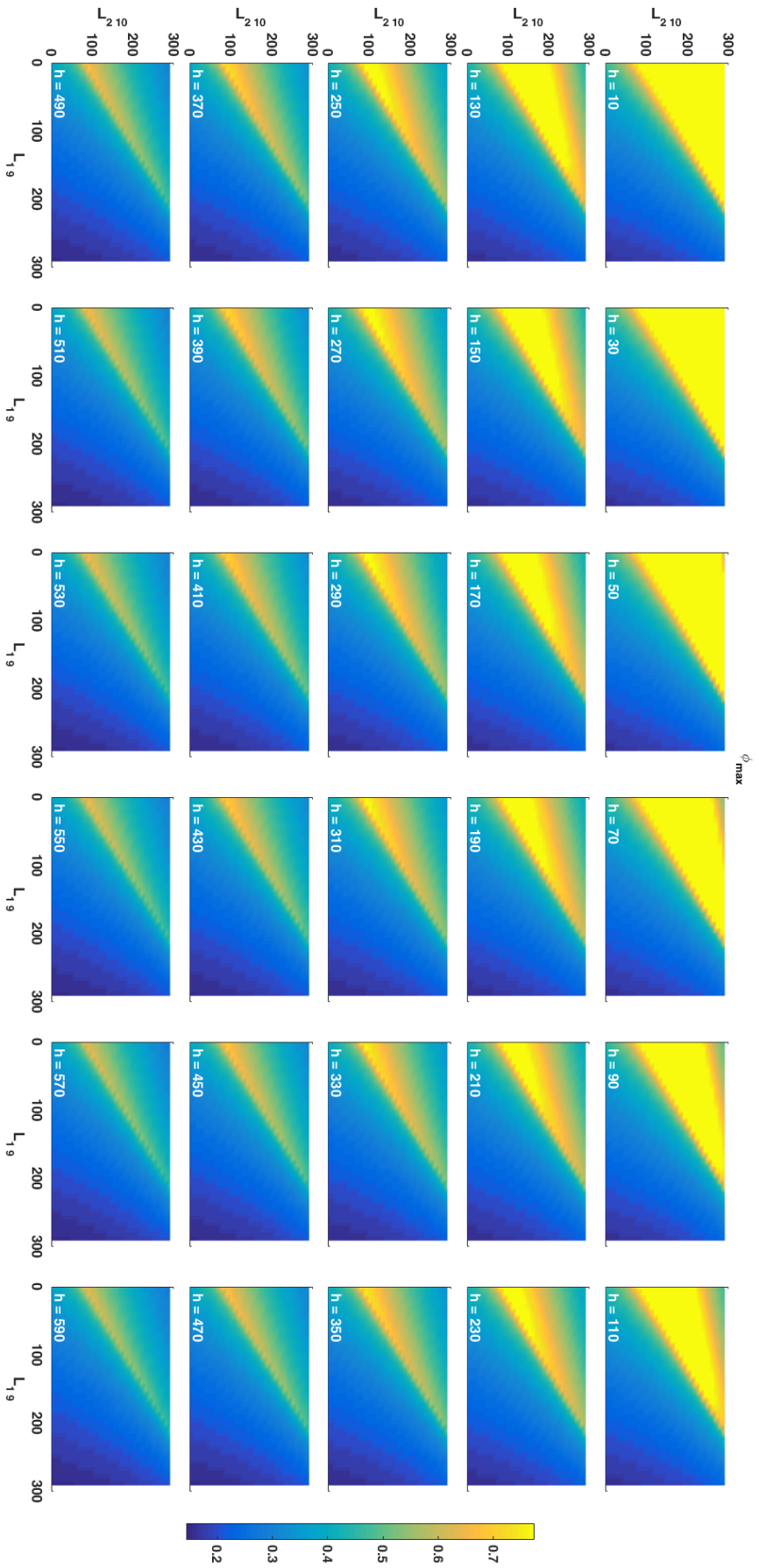


Figure 75: Value of ϕ_{max} in function of L_{19} (x axis), L_{210} (y axis), and h (different figures)

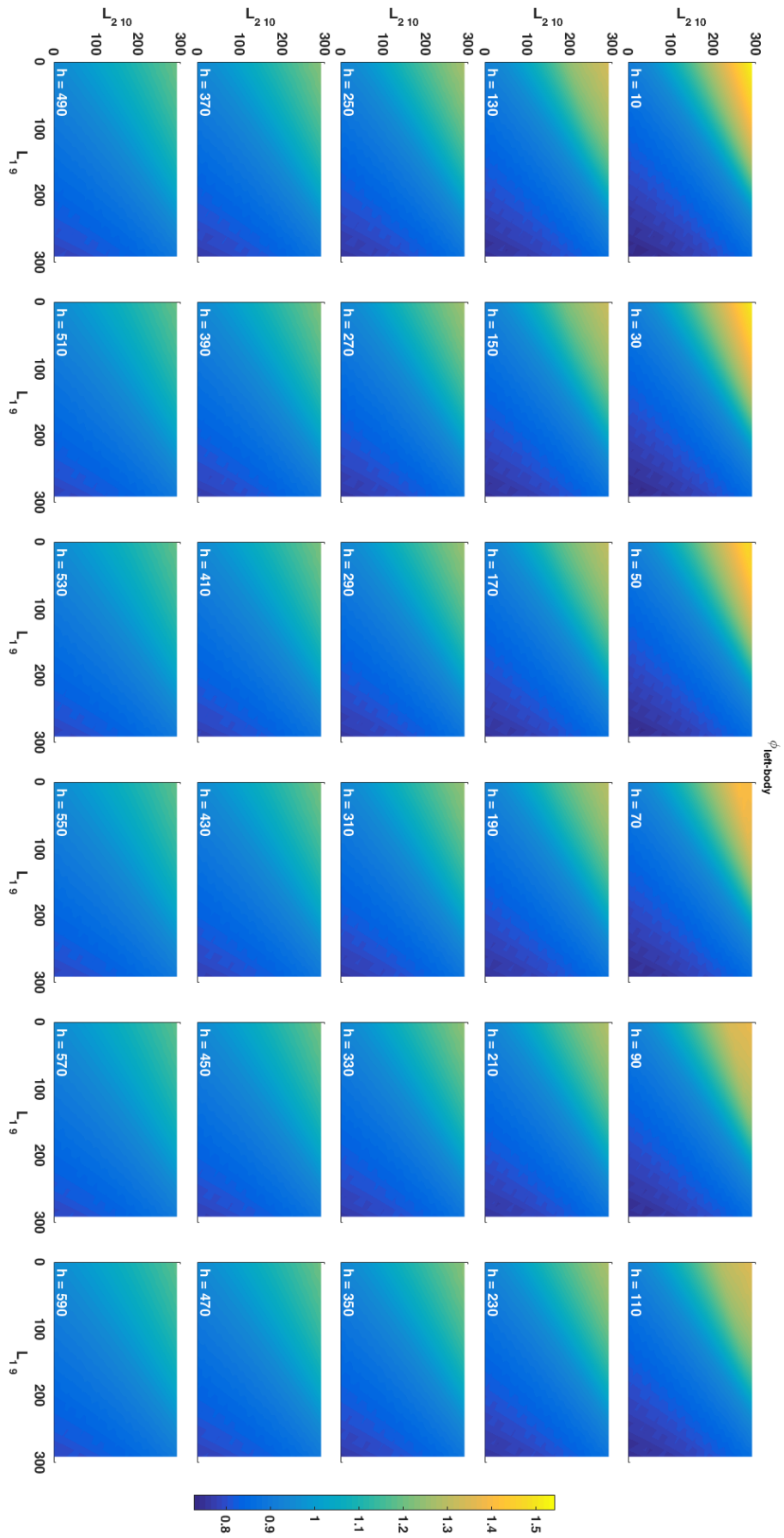


Figure 76: Value of ϕ_{inner} in function of L_{19} (x axis), L_{210} (y axis), and h (different figures)

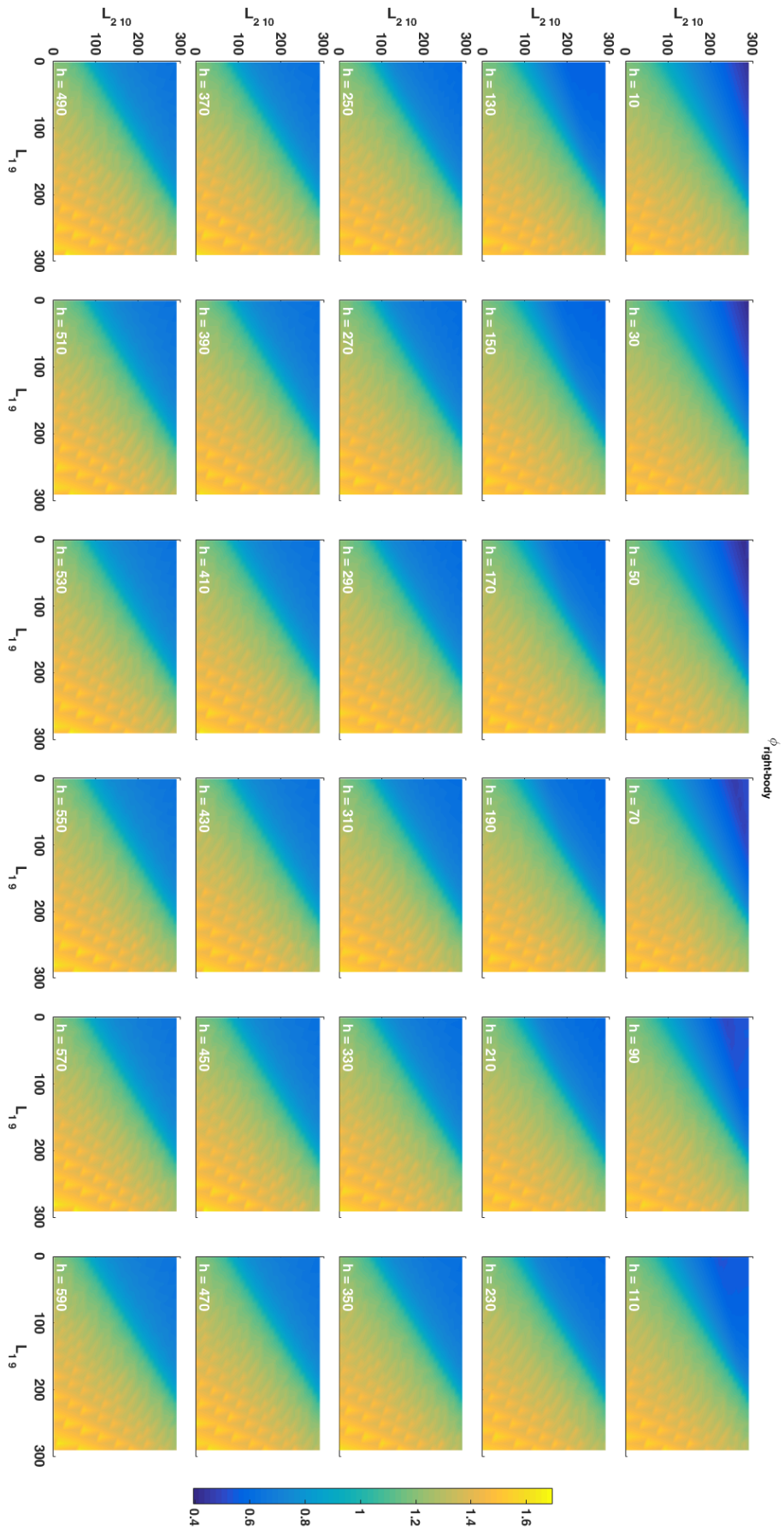


Figure 77: Value of ϕ_{outer} in function of L_{19} (x axis), L_{210} (y axis), and h (different figures)

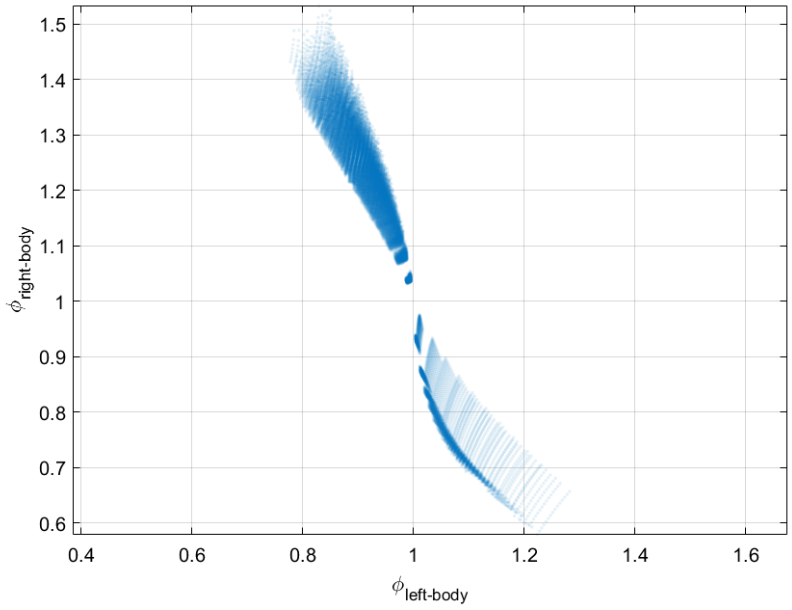


Figure 78: Ratio of angles on inclination $\phi_{inner\ to\ body}$ vs $\phi_{outer\ to\ body}$ in function of L_{19} (x axis), L_{210} (y axis). Each point represents a combination of the geometry

Dynamics

The dynamic analysis has been focused on the estimation of the necessary torque to maintain the body in vertical position. The worst case scenario for the motor is when the user gets on the vehicle. The motor will be able to return to the vertical position only if the torque requirement is lower than the maximum torque of the motor. On the contrary, inn a dynamic situation the forces on the vehicle will help the motor to return to the vertical position after a curve.

To carry out the dynamic simulation, the weight of each element is estimated with their material density and approximate dimensions. In addition, the weight of an standard (80kg) driver is include at a height on 1m (around the center of gravity).

Then, the model is forced to a known initial position (ϕ is the input) and the torque to return to the vertical position is calculated. For this step the expressions presented previously, based on the Lagrangian formulation, is used:

$$M_t = \lambda_m = -\Phi_g^T \begin{bmatrix} \lambda_1 & \lambda_2 & ... & \lambda_{20} \end{bmatrix}$$

The model was also slightly modified to include the tilting mechanism. The design B.4 was selected for this analysis, which consisted on a gear system attached between the motor and the lower suspension arms. Two different gear ratios were studied, with an increase of 1.5:1 and 4:1 in the motor torque:

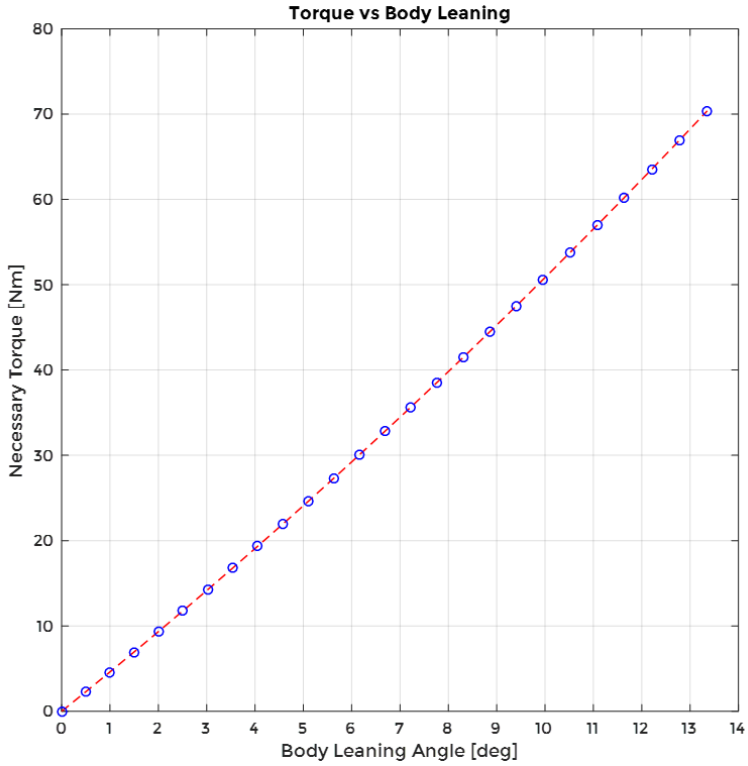


Figure 81: Necessary motor torque to restore vertical position with gear ratio=1.5

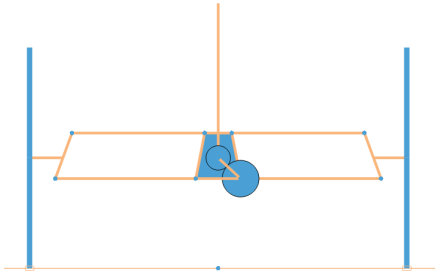


Figure 79: Studied model with gear ratio=1.5

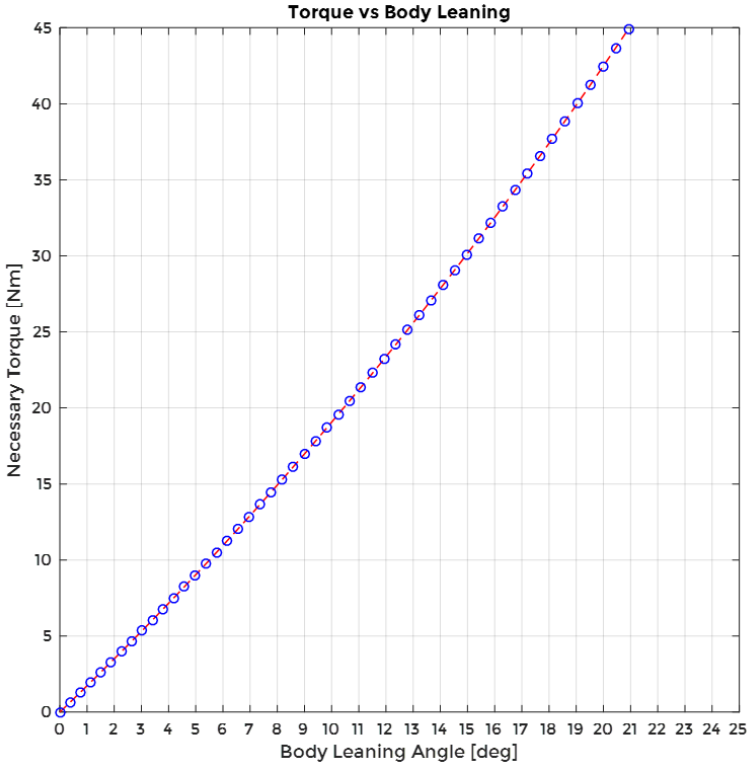


Figure 82: Necessary motor torque to restore vertical position with gear ratio=4

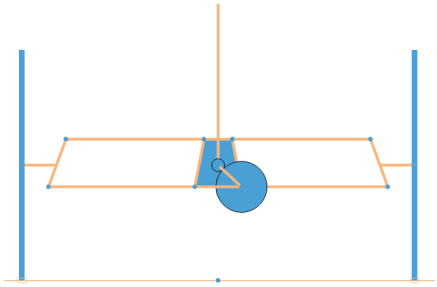


Figure 80: Studied model with gear ratio=4

For acceptable leaning angles (around 15°) there is torque requirement of 50Nm for the 1.5:1 gear ratio and of 15Nm for the 4:1 gear ratio. Let us study the characteristics of the available tilting motor to verify that this conditions are fulfilled.

NIDEC 48R Motor

The selected motor is a NIDEC 48R BLDC motor. Brushless DC motors do not require commutators and brushes, and have a long life, quietness, and high efficiency. It has a 144 ratio gear attached, which was customized for high-torque applications. The datasheet of the motor (without the gear) is included in the appendices. The rated power is 67W, with a nominal output torque T_r of 0.2Nm and a current consumption of I_r of 3.8A. In no load conditions, the output speed is $n_0 = 44601/min$, with a current consumption of 0.5A.

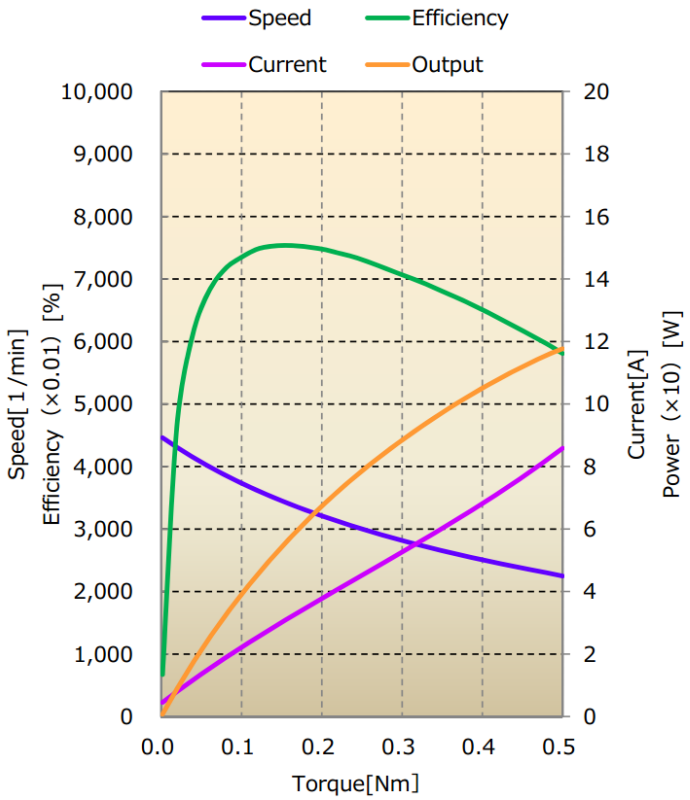


Figure 83: NIDEC 48R motor curves

Considering these two situations –nominal and no load–, the curve torque vs current can be obtained.

$$T = m\,I + n; \quad 0 = 0.5m + n; \quad 0.2 = 3.8m + n$$

$$T = \frac{2I - 1}{33}$$

With a output ratio of 144 in the planetary gear, and limiting the **maximum current to 5A**, the maximum torque is

$$T_{max} = 144\frac{25 - 1}{33} \approx 40Nm$$

With an additional gear, the torque increases to **60Nm and 160Nm with 1.5 and 4 gears ratio** respectively. Therefore, the **motor is able to recover** from the considered situations. In the view of the results of the dynamic analysis, the gears between the motor and the suspension arms will be designed to have a **ratio of 2:1**.

Design

The previous simulations have offered a better understanding of the motion of the front suspension and have helped to complete its geometry. In this section we will go through the design process for the different parts of the PEV.

Tilting

For the tilting part some components were reused from other vehicles (wheels, suspension arms and hubs). The wheels are not perfectly suited for tilting vehicles, due to their narrow width. On the other side, the hubs determine the height of the suspension points in the frame, since in steady position the suspension arms are intended to remain parallel to the ground.

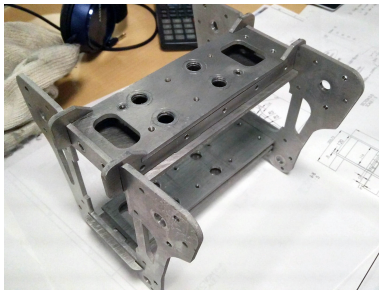


Figure 84: Aluminum part to connect the frame and the suspension arms. The geometry fits with the output from the kinematic simulations (part upside down)

The key point in the design of the tilting suspension was the selection of ball joints as articulations. Super-swivel ball joint rod ends were selected due to the 55° angle of ball swivel, being able to accommodate more misalignment than any other externally threaded rod end. This high angle allowed the tilting of the vehicle up to 30°.

The rest of the joining components were purchased at McMaster.com –their cost has been summarized in the Ch.7: Cost Summary–, and their datasheets have been included in the appendices. The joining of the suspension arms and the frame was carried out by a aluminum part made of water jetted sheets of 1/8'=3.125 mm. This part followed the geometry extracted from the MATLAB simulations, having 60 mm and 100 mm between the upper and lower suspension points respectively.

Figure 85: Initial state of the PEV frame

The gears were designed using a design table from excel[63] that was then imported to Solidworks. This table took into account all the parameters of the gears. As stated previously, the ratio of the gears was 2:1, with a width of 1/4'=6.35 mm and were fabricated with the water jet machine. The gears were inserted into the suspension arms by cutting them in half. This issue was due to the fact that the suspension arms were already fabricated and it was not possible to insert the gears in their position without breaking some parts.

[63] Solidworks Gear Generator. Available at https://www.thingiverse.com/thing:8077

Half Width	h	0.785
Addendum	a	1
Dedendum	b	1.25
Fillet Radius	e	0.38
Module	m	1
Teeth	z	20
Profile Shift	s	0.25
Pressure Angle	α	0.349
Pitch Radius	R	10
Base Radius	R_0	9.397
Addendum Radius	r_a	11
Half Angle	γ	0.093
Fillet Center	v_C	-0.87
Fillet Center	u_C	1.506

Table 6: Gear parameters

But far from meaning a problem, being able to put and remove the gears allowed to test the vehicle with and without tilting90. This modular advantage probed to be really useful, disassembling the gear and replacing a pair of shock absorbers allowed to test the vehicle with no tilting in a very straightforward way.

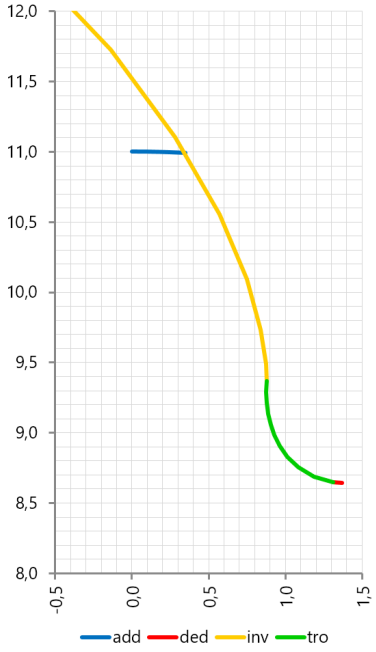


Figure 86: Gear Design: Addendum, Involute, Trochoidal and Dedendum sections

The frame was first modelled in Solidworks, as the baseline of the rest of the components. The non relevant parts were imported from online resources as GradCad, for example, the wheels. Once the frame was in Solidworks, the rest of the parts were designed and fabricated in the Media Lab's machine shop.

The assemble of the parts did not imply any problem. At this stage there was not steering system, so the wheels could rotate freely. This complicated a bit the tilting tests, since the wheels started rotating when the suspension moved. Nevertheless, the tilting mechanism worked satisfactorily and the inclination angles were as high as expected.

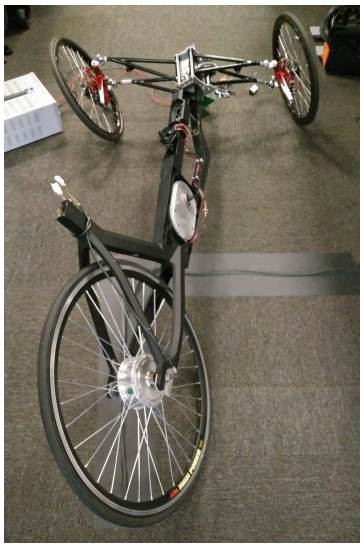


Figure 87: Top view: PEV tilting, first test without steering

The NIDEC motor was controlled with a driver connected to the Arduino board. The control strategy uses a PWM signal to command the speed of the motor and the position is controlled with a PID. In the electronics section a deeper explanation is included.

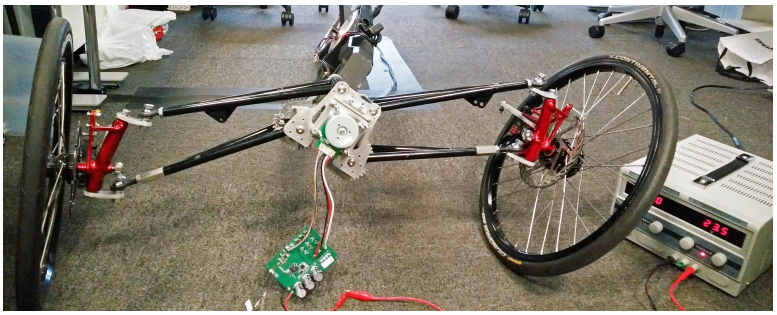


Figure 88: Front view: PEV tilting, first test without steering

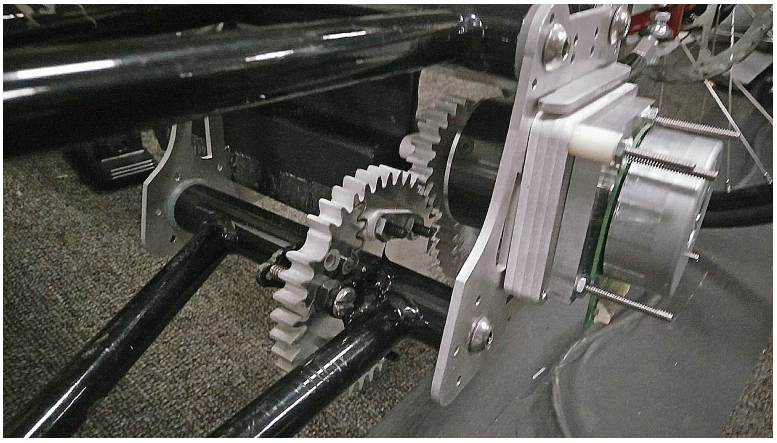


Figure 89: Tilting mechanism: motor and gears

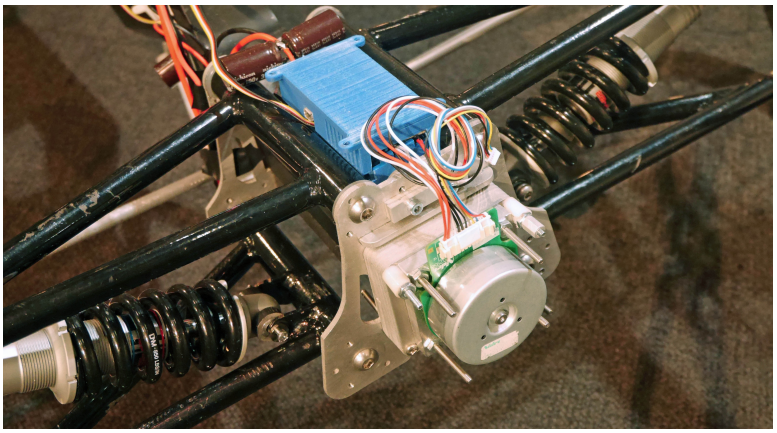


Figure 90: Front suspension without tilting

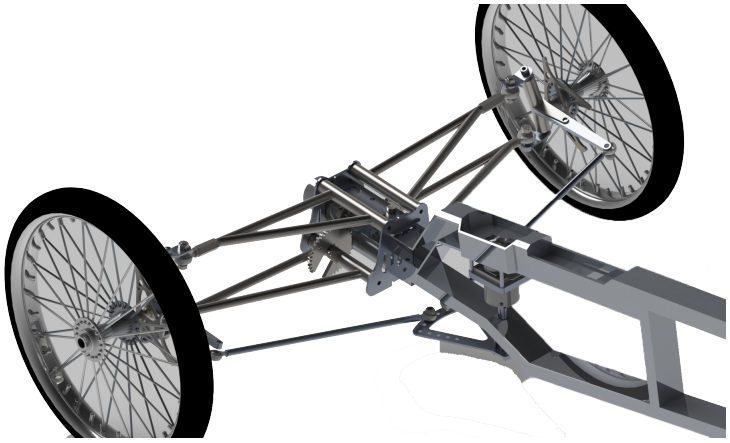


Figure 91: Render of the front suspension

Steering

During the fabrication process it was decided to implement a steer-by-wire system. There were two main reasons to justify this system:

- **SDTC**: by introducing a motor to control the steering mechanism, there is the possibility of changing the steer angle from the driver. The SDTC control strategy requires the control of the wheels to modify the path of the vehicle and tilt the vehicle by countersteering at high speeds. Even though in this project there was no time for implementing the SDTC, the steer-by-wire system remains useful for future applications of this vehicle.

- **Autonomy**: an actuated steering system is a basic feature for an autonomous lightweight vehicle. If a fleet of PEV is on the streets and a user calls one of them, it will need of a motorized propulsion and steering as well. In a reduced manner, in this project it was possible to remotely control the PEV and test it without a driver.

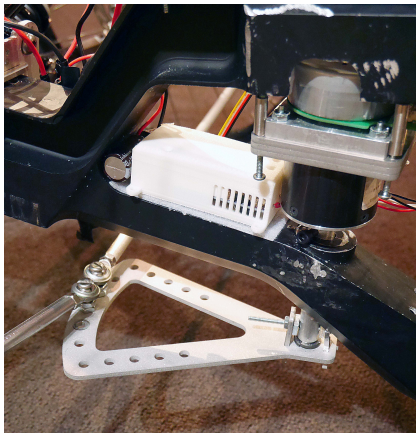


Figure 92: Steering mechanism, NIDEC motor and VESC controller

The main drawback of the steer-by-wire system is that it requires to be constantly powered by the batteries. When the vehicle is powered off, the steering motor remains in the same position, until is powered again and the handle bar and the wheels align.

Arduino boards do not have save any data after they are powered off. This fact implies that if the handle bar is moved during a no-power period, there will be a misalignment between the wheels and the handle bar. To avoid this problem, the absolute orientation of the IMU is used. By configuring the IMU properly, the orientation with respect to an inertial frame can be obtained. In case that the vehicle is rotated, there will also be a misalignment again, so two identical IMUs were necessary. The relative angle between them will report to the motor the initial angle of the wheels.

In the Figure 92, the position of the motor and the steering mechanism is represented. Some modifications were made to the aluminum frame to allocate the motor. The output shaft goes through the frame, connected to another aluminum part. This part (Figure 93) had several holes to adapt to the rest of attachments. The white case was 3D printed and contains the VESC controller.

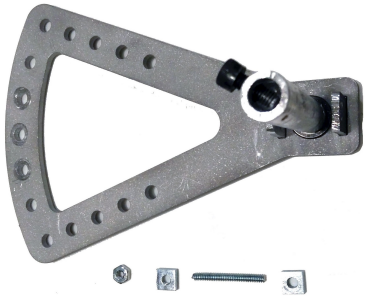


Figure 93: Steering parts

Handle Bar

Before approaching the design of the handle bar, it was necessary to build a column that supported it. The frame did not have any prepared holes or location for the steering column, so everything had to be designed from scratch and adapted to assure a good joining between the frame and the steering column.

Figure 94: Handle bar and steering column render

The steer-by-wire system allowed a free design of the column, formed by two lateral aluminum sheets connected by thinner transversal sheets. These parts were water jetted, so a distinctive shape was selected for them. While the fabrication resulted really fast and easy, assembling all these parts required some time. The assembly was really tedious, requiring some clamps to put all the parts together (Figure 95).

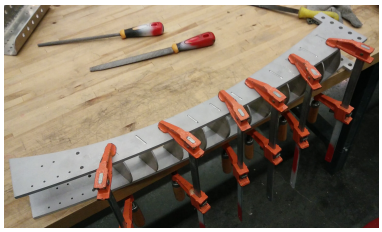


Figure 95: Assembly of the handle bar column

At the bottom of this column there was enough room for allocating the wiring and the Arduino boards. The attachment to the frame was done with a bended aluminum part. It is important to point out that it was decided to fabricate the PEV without using the milling machine. Apart from the time that the milling process takes by itself, a training and a preparation of the necessary files for its production were necessary.

On top of the column another NIDEC motor was assembled; its shaft had the handle bar connected (Figure 96). The reasons to introduce another motor in the handle bar are summarized as follows:

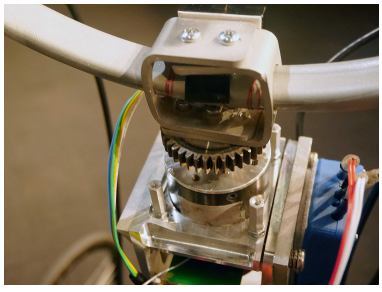


Figure 96: Handle bar: motor joining

– **Haptic Feedback**: a fundamental part of the steer-by-wire system is to give feedback to the driver about the forces that the steering motor is withstanding. In this way, the driver will have a subconscious input about the forces required to move the wheels, and the user experience will be enhanced.

This is also important in term of safety. Moving the handle bar without any friction can be dangerous, since the driver can make sudden turns, leading to a crash or fall.

The implemented feedback was not a realistic input from the forces happening in the steering motor. Instead, the force to move the handle bar linearly increased with the steering angle and the vehicle velocity. There was also a limit in the maximum input angle, generating enough force to block any motion. To do so, the motor variables read from the VESC controller were used, mainly the battery current, the motor current and the tachometer.

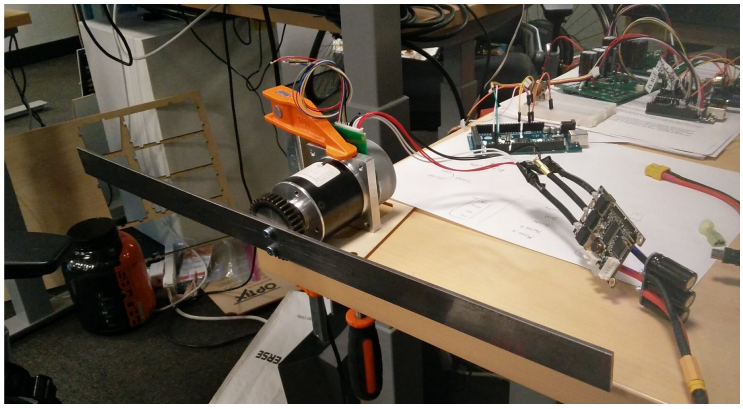


Figure 97: Test bench for the haptic feedback

At low speeds, the angle range was quite wide –but limited– and the forces required to move the handle bar were low. At higher speeds, the handle bar was very limited to a low range of angles and also it required a lot of force to move the handle bar. This helped to reduce the speed of steering, and protected the user to exceed the steering input at high speeds.

– **Alert/notify user**: Apart from controlling the steering input from the driver, a motorized handle bar can warn the driver about an obstacle in the road or any other issues with the vehicle or the road conditions. Vibrating the handle bar can the fastest way to notify the user about any problem, reducing the time of reaction and thus protecting the user.

– **Compass**: A possible scenario when riding the PEV can be the indication to go right or left when an address has been indicated to the system. Sometimes when riding a bike in a new city or in a unknown neighborhood it is necessary to stop an check the map to orient ourselves and find the next spot.

The handle bar contains a grip and a brake on each side and a potentiometer on the right side (Figure 98). The left and right brakes activate the brakes on the left and right front wheels respectively, and the potentiometer activates the power assist on the rear motor. A higher value of the potentiometer means a higher duty cycle in the rear motor, thus increasing the velocity of the vehicle.

Figure 98: Handle bar

a) Left grip and brake b) Right grip, brake and throttle

Driver ergonomics

The height and the position of the handle bar was designed to follow the indications in the book "The Guide to Cycling Ergonomics" by Ergotec[64]. The angles of the driver's articulations have been included in the drawings appendix.

[64] Ergotec. *The Guide to Cycling Ergonomics*. Wilhelm Humpert GmbH Co., 2012. Available at http://www.99.ucc.ie/ergonomics/T1.pdf

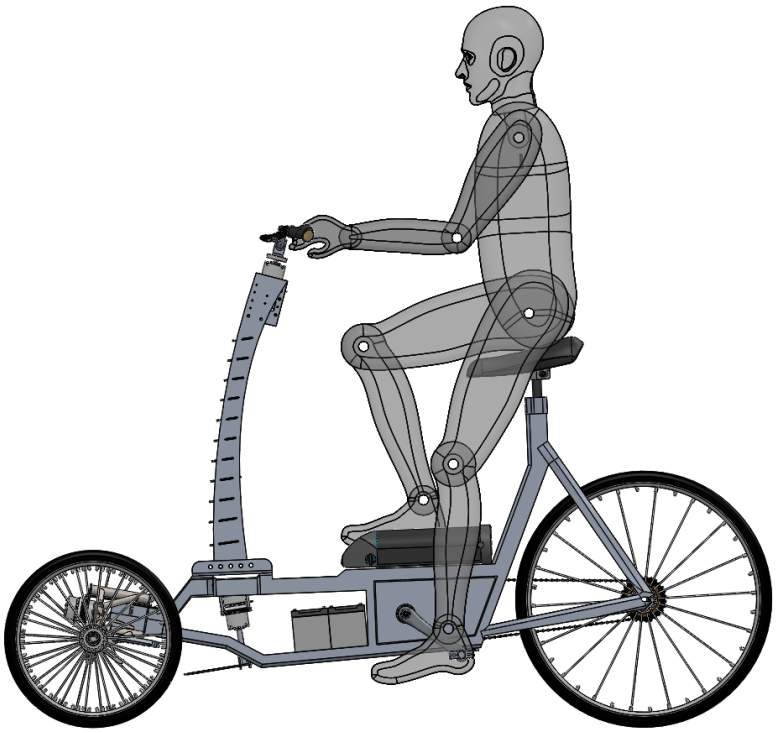


Figure 99: Drivers position on the CAD model

Power Assist

The PEV is power assisted by a brushless motor located in the rear wheel's hub. This motor is a 36V E-Bikeling 500W geared motor[65]. It provides enough power to assist the propulsion of the PEV. The system comes prepared to attach a sprocket to the hub, so that the rear wheel can be moved with the pedals. All the drivetrain parts were mounted in the frame.

[65] DC motor in rear wheel hub. Available at http://www.ebikeling.com/shop/electric-bicycle-36v-500w-geared-rear-26-kit-opt

The rear brushless motor, as every NIDEC motor (tilting, steering, handle bar), are controlled by VESC. In the next section –electronics– more detailed information is included. The PEV is intended to be a lightweight vehicle fully prepared to incorporate a modular autonomous package. This kit will transform a normal three wheeler vehicle into an autonomous urban vehicle. That is why a motor to propulse the vehicle is necessary, as well as another motor to steer the front wheels.

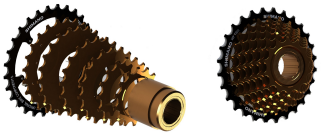


Figure 100: PEV sprocket

The motor will assist under the user demands. In the right side of the handle bar there is a potentiometer that will throttle the vehicle when turned on. Regarding the signal flow, the potentiometer sends the command to the Arduino Mega, which is connected to the VESC controller through serial. The VESC finally controls the motor and moves the rear wheel.

Figure 101: Rear motor in the hub

Figure 102: PEV Pedal system: chain, gears and pedals

Figure 103: Isometric render

Figure 104: Lateral render

Figure 105: Front render

Figure 106: PEV components explosion

Figure 107: PEV

Electronics

Electronics lay the foundation for a mechatronical project. In this section the used components are presented and the process to implement them correctly in the PEV is explained.

PID Control

Before using the VESC controller –open source, highly modifiable electronic speed controller ESC by Benjamin Vedder–, a NIDEC driver was used to control the angular position of the tilting motor. This happened in an early stage part of the project, so that the PEV final version was designed with VESC controllers.

The driver had 4 pins to control the motor:

- **PWM**: Pulse width modulation signal, it determines the speed of the motor. It is considered as a integer between 0 and 255. The higher PWM, the faster the motor rotates. Connected to a PWM digital output pin in Arduino.

- **CCW**: Binary variable to select the direction of rotation (clockwise or counter clockwise). Connected to a digital pin.

- **FG**: Frequency Generator, it outputs a signal with a frequency proportional to the motor speed. The proportional constant was unknown, it was roughly estimated.

- **GND**: ground connected to the GND port in Arduino.

This driver was used to control the angular position of the tilting motor. The input to the motor will be the value of PWM, that will make the motor rotate at a certain speed. The speed will be read by the pulses coming through the FG pin. Since there are not position readings, the position will be calculated based on the speed and the frequency of the control.

Before reviewing the control system in detail, it is important to make a distinction between the temporal and the Laplace domains. The control diagram represents the Laplacian domain, where the time variable t is replaced by the complex variable (frequency s) by applying the Laplace transform. A variable in lowercase (n) will belong to the temporal domain, whereas a variable in uppercase (N) will belong to the Laplacian domain.

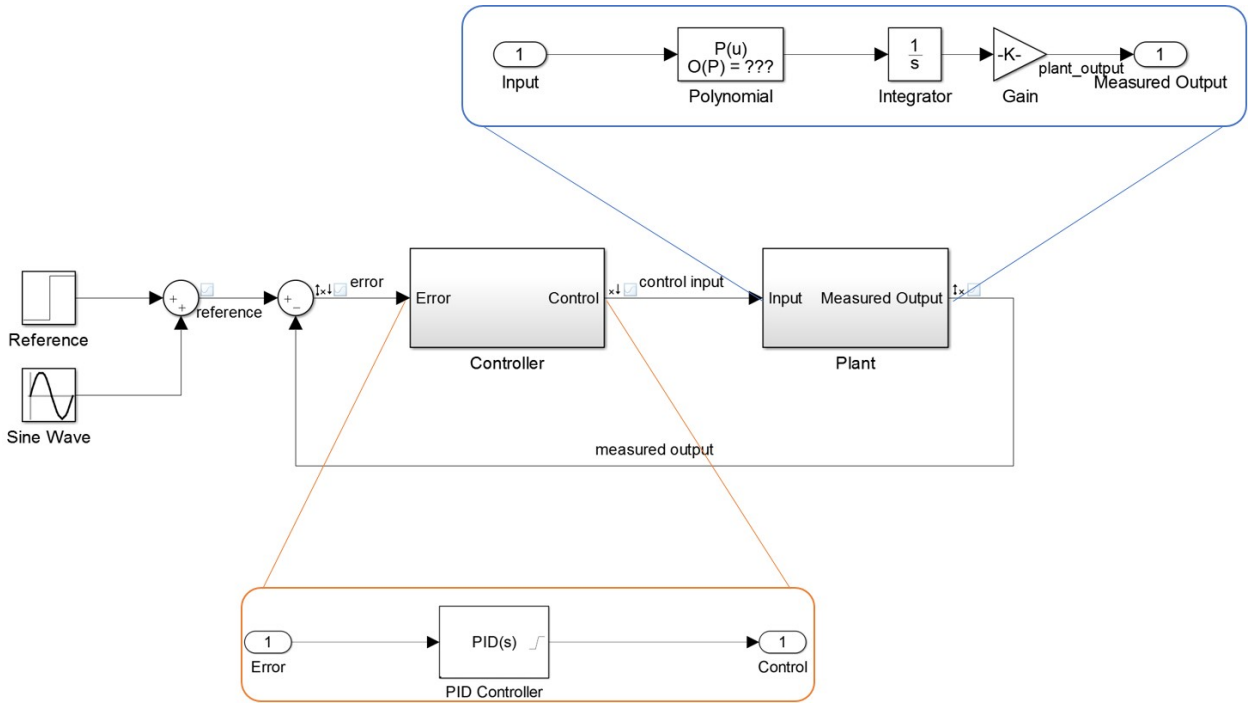


Figure 108: PID control of the angular position

The control diagram in Figure 108 represents the inputs and outputs of this simple position control. The angular reference ϕ_{ref} is given by the user – at first through a potentiometer– and the error e is calculated between the reference and the actual angle ϕ. This error will be used to determine the input signal pwm using a PID control. Finally, the pwm will put an specific speed and therefore a position in that timestamp.

Translating these words into equations, the system equation is expressed as

$$\phi_i = \phi_{i-1} + 60n(t_i - t_{i-1})$$

where ϕ_i is the angular position at time t_i and n is the speed in rpm.

The input variable pwm is calculated from the error (defined as $E = \Phi_{ref} - \Phi$) by means of the PID control:

$$pwm = K_P(\phi - \phi_{ref}) + K_I \int_{t_{i-1}}^{t_i} (\phi - \phi_{ref})dt + K_D \frac{\partial(\phi - \phi_{ref})}{\partial t}$$

In the Laplacian domain:

$$PWM = (K_P + \frac{K_I}{s} + K_D)(\Phi_{ref} - \Phi)$$

The speed is the time derivative of the position with respect to the time:

$$n = \frac{\delta\phi}{\delta t} \frac{1\,rev}{2\pi\,rad} \frac{60}{1\,min} \quad \rightarrow \quad N = s\,\Phi \frac{60}{2\pi}$$

The relation between the input pwm and the output speed n is not known, but it can be estimated. Using the speed information from the FG pin, some measurements were made to characterize the speed of the motor in function of the pwm input. In Figure 109 the obtained data is represented and fitted into a 5th grade polynomial curve.

$$N = f(PWM)$$

In Figure 110, on the contrary, the speed-pwm relation is simplified by a linear regression (the speed is saturated at $pwm = 57$)

$$N = K\ PWM$$

Therefore, the speed is represented in function of the error:

$$N = s\,\Phi\frac{60}{2\pi} = f(PWM) = f\big((K_P + \frac{K_I}{s} + K_D)(\Phi_{ref} - \Phi)\big)$$

For the sake of simplicity, the transfer function is indicated for the relation $N = K\,PWM$

$$\Phi = \Phi_{ref}\frac{K_P\,s + K_I + K_D\,s^2}{\frac{60s^2}{2\pi K} + K_P\,s + K_I + K_D\,s^2}$$

The control strategy was implemented in MATLAB and the PID tuner turned out to be really useful to determine the values of K_P, K_I and K_D. Overall the quality of this **PID control was satisfactory**, but the VESC controller resulted easier to implement and besides it provided more feedback from the motor (electrical and mechanical variables).

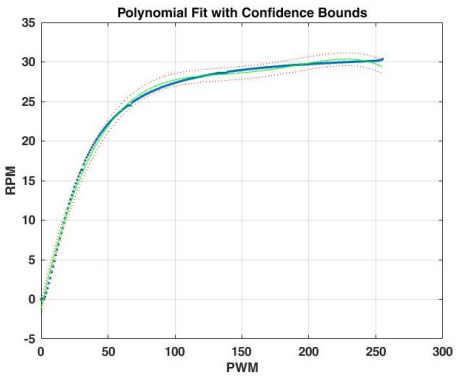


Figure 109: Angular speed N vs signal PWM: Polynomial fitting

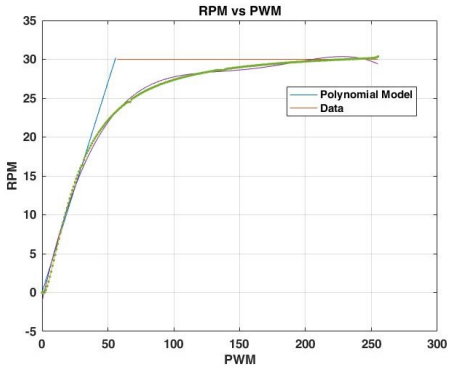


Figure 110: Angular speed N vs signal PWM: Saturation and linear fitting

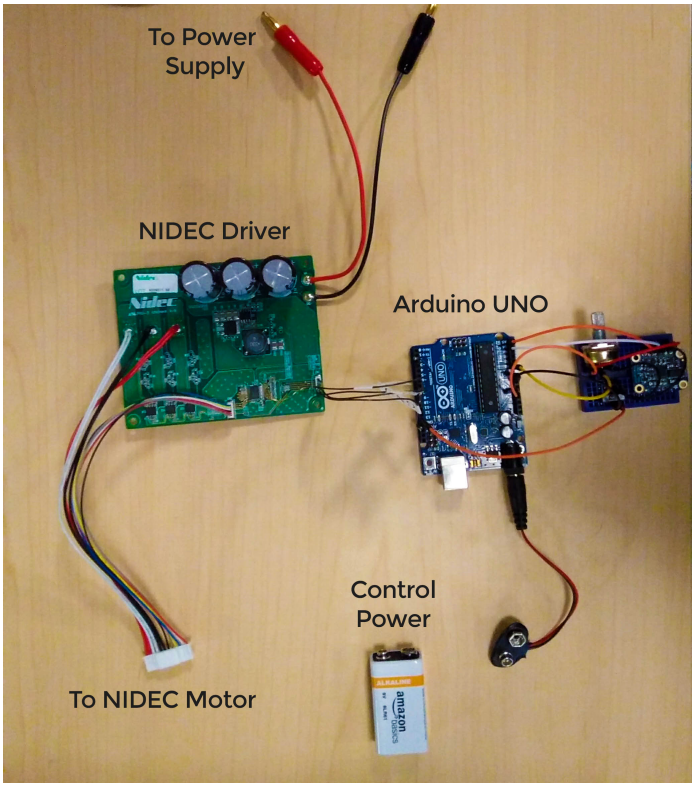


Figure 111: Schematic of the early motor controller: Arduino UNO, 9V battery, potentiometer and NIDEC driver

VESC

The VESC is a fully customizable open source electronic speed controller by Benjamin Vedder, designed for lightweight and compact applications like skateboards, or in this case, a three wheeler vehicle.

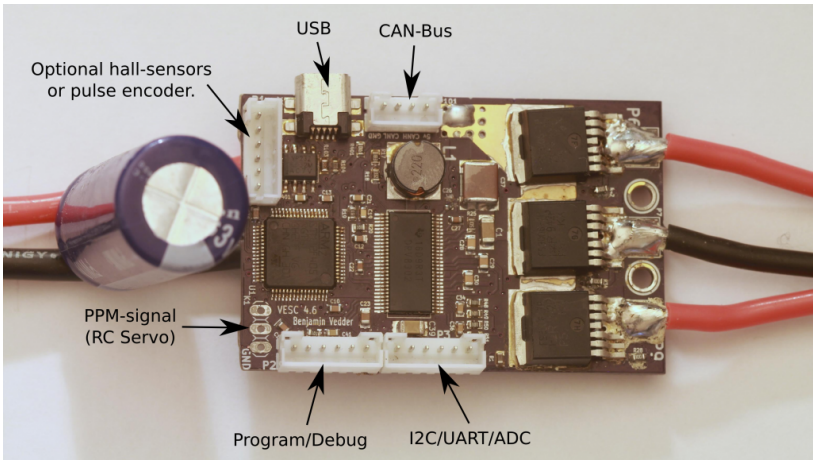


Figure 112: VESC Picture

The controller receives the power from the batteries and outputs the 3 phases U, V and W to the brushless motors. The incorporated firmware offers various possibilities to control the motor in different ways. Every motor in this PEV was configured to be controlled in FOC (Field Oriented Control) sensorless mode.

FOC – Field Oriented Control

BLDC motors require a controller that converts the applied DC from the battery cells into AC to drive the motor. This task demands complex driving algorithms to commutate the coils in a sequence that achieves the desired directional rotation. A wide range of control algorithms are available:

- Trapezoidal control: For each of the 6 commutation steps, a pair of windings are powered, leaving the third disconnected. This method generates high torque ripple, leading to vibration, noise, and poor performance.

- Sinusoidal control: it supplies sinusoidal varying current to the 3 windings, thus reducing the torque ripple and offering a smooth rotation. However, these time-varying currents are controlled using basic PI regulators, which lead to poor performance at higher speeds.

- Field Oriented Control: the torque and the flux can be controlled independently and provides faster dynamic response. There is no torque ripple and smoother, accurate motor control can be achieved at low and high speeds.

VESC Library

The VESC was connected to the Arduino MEGA board through serial connection (UART). To facilitate the control of the motor, the VescUartControl library was used in Arduino to interface over UART with the VESC.

The margin table summarizes the available data from the VESC. In addition, it was possible to set the motor current, the brake current, the angular position, the duty cycle and the RPM of the motor.

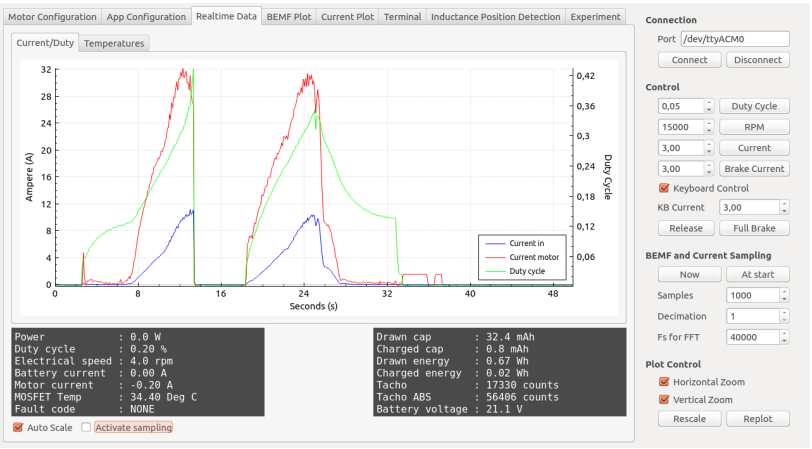
bldcMeasure struct
Average Motor Current
Average Input Current
Duty Cycle
Motor RPM
Input Voltage
Amperes Hours
Amperes Hours Charged
Tachometer
Tachometer Absolute

```
bool VescUartGetValue(struct bldcMeasure& values, int num);

        Sends a command to VESC and stores the returned data

        values(struct bldcMeasure&) — bldcMeasure struct with received data
        num(int) — the serial port in use (0=Serial; 1=Serial1; 2=Serial2; 3=Serial3;)
        return(bool) — true if success

void VescUartSetCurrent(float current, int num);

        Sends a command to VESC to control the motor current

        current(float) — the current for the motor
        num(int) — the serial port in use (0=Serial; 1=Serial1; 2=Serial2; 3=Serial3;)

void VescUartSetCurrentBrake(float brakeCurrent, int num);

        Sends a command to VESC to control the motor brake

        brakeCurrent(float) — the current for the brake
        num(int) — the serial port in use (0=Serial; 1=Serial1; 2=Serial2; 3=Serial3;)

void VescUartSetPosition(float position, int num);

        Sends a command to VESC to control the motor position

        position(float) — the position in degrees for the motor
        num(int) — the serial port in use (0=Serial; 1=Serial1; 2=Serial2; 3=Serial3;)

void VescUartSetDuty(float duty, int num);

        Sends a command to VESC to control the motor duty cycle

        duty(float) — the duty cycle for the motor
        num(int) — the serial port in use (0=Serial; 1=Serial1; 2=Serial2; 3=Serial3;)

void VescUartSetRPM(float rpm, int num);

        Sends a command to VESC to control the motor rotational speed

        rpm(float) — the revolutions per second for the motor
        num(int) — the serial port in use (0=Serial; 1=Serial1; 2=Serial2; 3=Serial3;)
```

VESC Features

- Voltage $8V$ to $60V$
- Current up to 240A for a some seconds or 50A continuous
- 5V 1A output for external electronics (arduino)
- Sensored and sensorless FOC, BLDC, and DC
- Current and voltage measurement on all phases
- Duty-cycle control, speed control or current control
- Interface to control the motor: PPM signal, analog, UART, I2C, USB or CAN-bus.
- Regenerative braking
- Good start-up torque in the sensorless mode
- The motor is used as a tachometer, which is good for odometry
- Adjustable protection against low/high input voltage and high motor/input current.

It is possible to plot the currents in the BLDC tool, voltages and the duty cycle in real-time. This is useful when debugging how everything behaves. Some screenshots of the configuration GUI (BLDC Tool):



In order to protect the VESC from hazard and avoid any undesired contacts, some cases were 3D printed. The first version was designed to allocate only the board, whereas the second version had enough space to also incorporate the VESC capacitors.

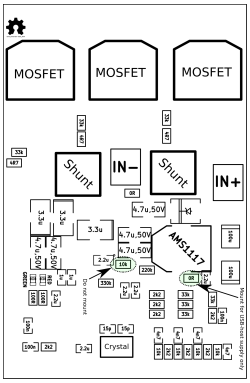


Figure 113: Front VESC Schematic

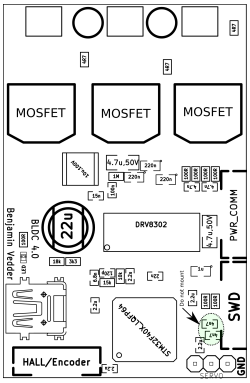


Figure 114: Back VESC Schematic

Figure 115: BLDC Tool

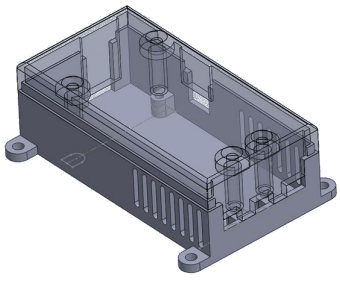


Figure 116: VESC case version 1

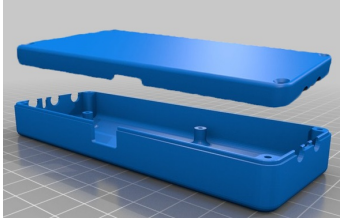


Figure 117: VESC case version 2

Rotary Encoder

The longitudinal velocity of the PEV $V = V_x$ is obtained from a rotary encoder located in the rear wheel. This sensor is completely necessary, since the motor is not able to provide this information. The motor and the hub of the rear wheel are disentangled, meaning that the motor freely rotates inside the hub when the user pedals, for example. That is why the motor cannot give a feedback of the velocity of the wheel.

The rotary encoder or transmitter[66] is a incremental optical encoder, that converts the motion to an electrical signal to indicate the position of the rear wheel. Transmitters must be used with a controller that has quadrature detection to get a 4X resolution increase and meet IP50 for protection from dust.

[66] Rotary Motion Position-Measuring Transmitter. Available at https://www.mcmaster.com/#16695t57/=17znmw6

The code disk inside a quadrature encoder contains two tracks usually denoted Channel A and Channel B. These tracks or channels are coded ninety electrical degrees out of phase and this is the key design element that will provide the quadrature encoder its functionality. In applications where direction sensing is required, a controller can determine direction of movement based on the phase relationship between Channels A and B. As illustrated in the figure below, when the quadrature encoder is rotating in a clockwise direction its signal will show Channel A leading Channel B, and the reverse will happen when the quadrature encoder rotates counterclockwise.

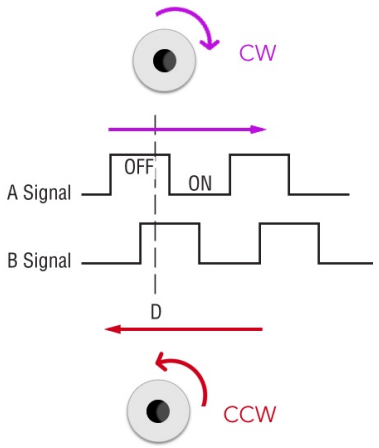


Figure 118: Quadrature Encoder

The resolution of this particular encoder is of 1000 counts per revolution. It has a 1.91"=48.5 mm diameter circumference polyurethane wheel attached to the shaft of the encoder. To get the speed of the PEV, a simple kinematic study was done, following the diagram in Figure **??**:

$$\vec{v}_0 = 0 \quad \vec{v}_A = V\,\vec{i} = w_1\,R_1\,\vec{i} \quad \vec{v}_P = \vec{v}_A + w_1\,R_1$$

$$\vec{v}_B = V\,\vec{i} = w_2\,R_2 \quad \vec{v}_P = \vec{v}_B + w_2\,R_2$$

Equaling the expression for the P point velocity $\vec{P}$:

$$\vec{v}_A + w_1\,R_1 = \vec{v}_B + w_2\,R_2$$

Since the velocities in the centers of both solid is the same ($\vec{v}_A = \vec{v}_B$)):

$$w_1\,R_1 = w_2\,R_2$$

and therefore,

$$V\,\vec{i} = w_1\,R_1 = w_2\,R_2$$

For calculating the speed of the vehicle, the angular velocity and the radius of the small circumference attached to the encoder is only needed.

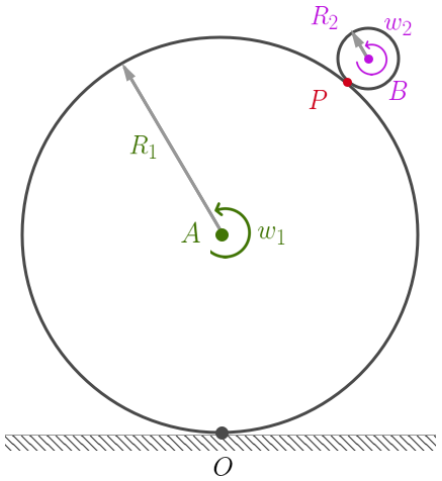


Figure 119: Wheel – Encoder diagram

The radius is known, and the angular velocity can be defined as:

$$w_2 = \frac{\Delta\phi}{\Delta t}; \quad \Delta\phi = N_{counts}\frac{1\,rev}{1000\,counts}\frac{2\pi\,rad}{1}$$

The speed is therefore calculated as:

$$V = N_{counts}\frac{2\pi R_1}{1000\Delta t}$$

Every time that the encoder sends a pulse to the board, an interrupt routine starts to read that pulse and catch the rising or falling edge on the input pin. Therefore, if the Arduino is not fast enough, interrupting the microprocessor in the board can deny the reading of other pulses and considerably delay other operations.

Taking into account that the resolution was really high (1000 counts/rev), and a radius of the encoder was 48.5/2=24.25 mm if the PEV is going at $1m/s$, for example, that means that in 1 second, the Arduino board needs to be able to interrupt its code more than 6500 times (6500 Hz). Increasing the speed of the vehicle increases this frequency as well.

To get the number of counts in a given time interval, it was necessary to implement a very fast digital reader in the Arduino Mega in order to not lose any count and estimate correctly the speed of the vehicle.

In addition, this strategy works even better if a bit of hardware debouncing is forced on the rotary encoder. With two 0.1 uF capacitors soldered to the encoder pins, the number of calls to the interrupt routine is dramatically reduced. This is important because too many calls to the interrupt routine will rob computing cycles from the main routine, negating the effects of using interrupts to save computing power.

Figure 120: Rotary encoder mounted on the PEV

IMU

The selected inertial motion unit was the Bosch BNO055, a 9-axis absolute orientation sensor with sensor fusion. The BNO055 integrates a triaxial 14-bit accelerometer, a triaxial 16-bit gyroscope with a range of 2000 degrees per second, a triaxial geomagnetic sensor and a 32-bit microcontroller.

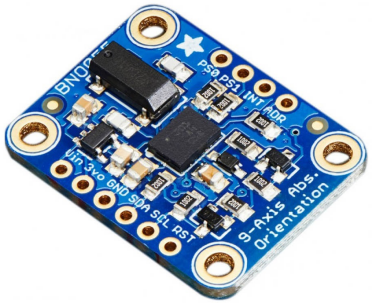


Figure 121: Bosch BNO055 9DOF IMU

VIN: 3.3-5.0V power supply input

GND: common for power and logic

SCL - I2C clock pin

SDA - I2C data pin

ADR: to change the I2C address

It is connected through I2C (clock and data pins) to the analog pins in the Arduino. The PEV is equipped with a pair of BNO055, both connected through I2C, with the difference that the second IMU's ADC pin is connected to the board as well. Setting the ADR pin to high changes the I2C address from the default (0x28 to 0x29).

Rather than spending a lot of time with algorithms of varying accuracy and complexity, the data can be extracted very easily thanks to the sensor fusion. However, it requires to configure it properly when powered.

1. The operation mode has to be setup to be able to configure it (CONFIG MODE)
2. Limit accelerometer range to 2G, get better accuracy
3. Calibrate using some offsets values obtained previously
4. The operation mode is setup again to Fusion mode with NDOF, mode from which the absolute orientation can be obtained.

Calibration

To obtain correct values from the IMU, it has to be properly calibrated first. Once the device is calibrated, the calibration data will be kept until the BNO is powered off. The BNO doesn't contain any internal EEPROM, so a new calibration will be needed every time the device starts up, or a manual restore of the calibration data.

The BNO055 includes internal algorithms to constantly calibrate the gyroscope, accelerometer and magnetometer. The four calibration registers – an overall system calibration status, as well individual gyroscope, magnetometer and accelerometer values – will return a value between '0' (uncalibrated data) and '3' (fully calibrated).

The sensors are trimmed to tight offsets, meaning valid data is obtained even before the calibration process is complete, but particularly in NDOF mode any data should be discard as long as the system calibration status is 0. The reason is that system cal '0' in NDOF mode means that the device has not yet found the north pole, and orientation values will be off. The heading will jump to an absolute value once the BNO finds magnetic north.

To generate valid calibration data, the following criteria should be met:

- Gyroscope: the device must be standing still in any position
- Magnetometer: normal movement of the device is sufficient
- Accelerometer: the BNO055 must be placed in 6 standing positions for +X, -X, +Y, -Y, +Z and -Z.

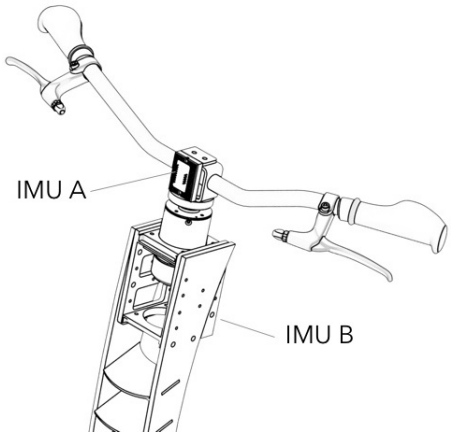


Figure 122: Location of IMU A and B

Position of the IMU

The PEV is equipped with a pair of IMUs, one placed on the handle bar and another one fixed on the frame. The justification for this decision lays on two arguments:

- The relative angle between the frame and the handle bar can be obtained, meaning that an encoder is not needed in the handle bar.
- The perceived lateral acceleration a_{per} has to be obtained along the lateral axis y', which should not be affected by the handle bar steering.
- The steering angle δ and its rate $\dot{\delta}$ can be also obtained from the gyroscope

Orientation of the IMU

Due to the fact that each IMU is oriented differently, it is necessary to process the orientation data depending on the case. The angular orientation from each IMU is obtained from the quaternion:

$$\vec{q} = \begin{bmatrix} q_w \\ q_x \\ q_y \\ q_z \end{bmatrix} = \begin{bmatrix} \cos(\alpha/2) \\ \sin(\alpha/2)\cos(\beta_x) \\ \sin(\alpha/2)\cos(\beta_y) \\ \sin(\alpha/2)\cos(\beta_z) \end{bmatrix}$$

where α is a simple rotation angle and $\cos(\beta_x)$, $\cos(\beta_y)$ and $\cos(\beta_z)$, are the direction cosines locating the axis of rotation.

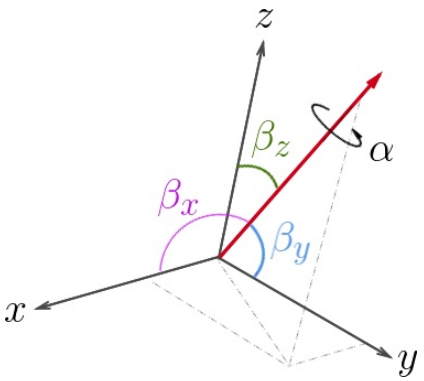


Figure 123: Quaternion vector diagram

The quaternion is then transformed into the appropriate Euler angles: Roll, Pitch and Yaw. Each of these angles is related to one of the axis X, Y and Z. Depending on the order of the rotations –XYZ and ZYX Euler angles are not the same– the values of the roll pitch and yaw changes. For transforming the quaternion into the corresponding Euler angles, the motion of each IMU has to be taken into account, as well as their relative orientation.

The handle bar and the frame IMU's were placed in different orientations:

- In the handle bar IMU the angle of interest is the steering angle δ around the Z axis. In order to obtain it independently from any other rotation, the quaternion is transformed into the YZX euler angles

$$\delta_A = \arctan \frac{-2(q_y\, q_z - q_w\, q_{x)}}{q_w^2 - q_x^2 + q_y^2 - q_z^2}$$

- In the frame IMU the angles of interest are the steering angle δ and the tilting angle θ around the Y and the X axis respectively. Therefore, the transformation is done to obtain the XZY angles

$$\theta = \arcsin\big(-2(q_x\, q_z - q_w\, q_y)\big)$$

$$\delta_B = \arctan \frac{2(q_x\, q_y + q_w\, q_z)}{q_w^2 + q_x^2 - q_y^2 - q_z^2}$$

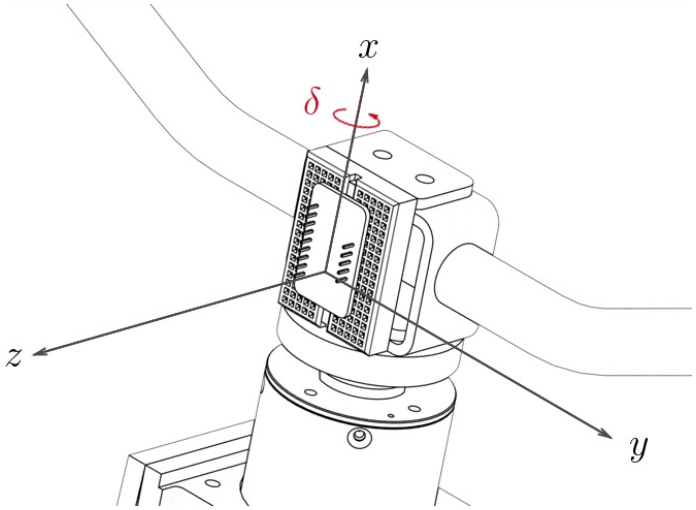


Figure 124: Handle bar IMU (A)

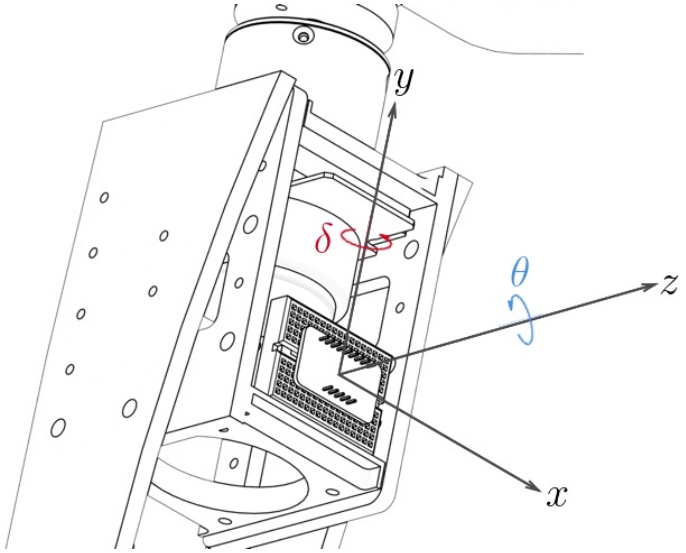


Figure 125: Frame IMU (B)

Extending angular range

The arctan and arcsin functions implemented only produce results between $-\pi/2$ and $\pi/2$. This range problem is solved by extending the angular orientation to a continuous spectrum. Any angle is defined as:

$$\varphi = \varphi + 2\pi k)$$

with $k = 0$ at the initial range $[-\pi/2, \pi/2]$.

- If φ exceeds a value of $(k+1)\pi/2$, then the k increments one unit: $k = k+1$
- If φ falls behind a value of $-(k+1)\pi/2$, then the k decreases one unit: $k = k-1$

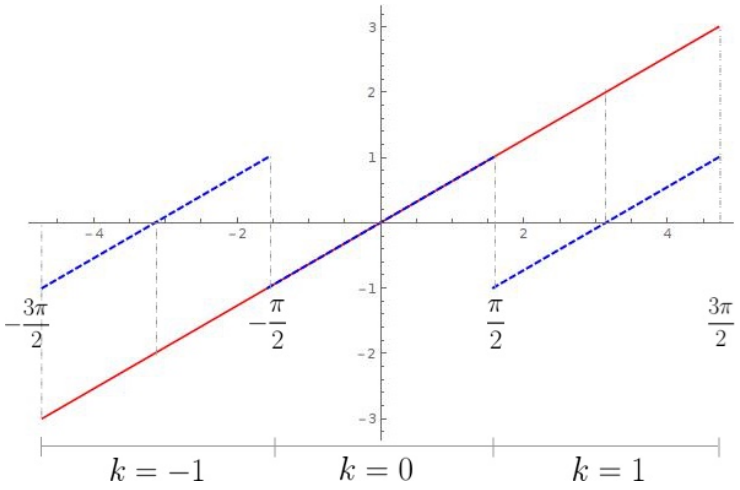


Figure 126: Discontinuous vs continuous orientation

Data obtained from the IMU

The control strategy requires some data to calculate the torque requirement from the tilting motor.

	Handle Bar IMU
δ_A	Euler angle along vertical direction
$\dot{\delta}$	Gyroscope along vertical direction

	Frame IMU
δ_B	Euler angle along vertical direction
$\dot{\Phi}$	Gyroscope along vertical direction
θ	Euler angle along longitudinal direction
$\dot{\theta}$	Gyroscope along longitudinal direction
a_{per}	Linear acceleration along lateral direction

The steering angle δ is calculated from the relative orientation $\delta = \delta_A - \delta_B$

Communication: Bluetooth Modules

The Arduino boards have two Bluetooth modules connected. The two modules, HC05 and HC06 are very similar. HC–05 is a more capable module that can be set to be either master or slave while HC–06 is a slave only device.

These modules run on 3.3V power and have two modes of operation. In command mode AT commands can be sent to it and in data mode it transmits and receives data to another Bluetooth module.

The HC–06 module was connected to an Android app in order to remotely control the PEV. The commands sent to this module controlled the rear motor, as well as the steering motor. The HC–05 module, on the other side, was connected to a Processing script, in which data was saved in real time and exported into a .csv file. In this way, it was possible to analyze the data coming from the sensors and verify the correct response of the control strategy.

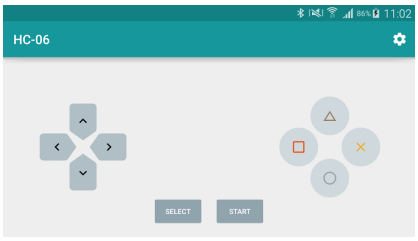


Figure 127: Android app for remote control

An application of this data sender module was used during the Media Lab's spring members week. The data from the PEV was streamed in real time and projected onto a table. Only basic data was presented during this event, for example, the speed of the rotary encoder and IMU data. The script was also prepared to align the projection to the table, as it can be seen in the Figure 128.

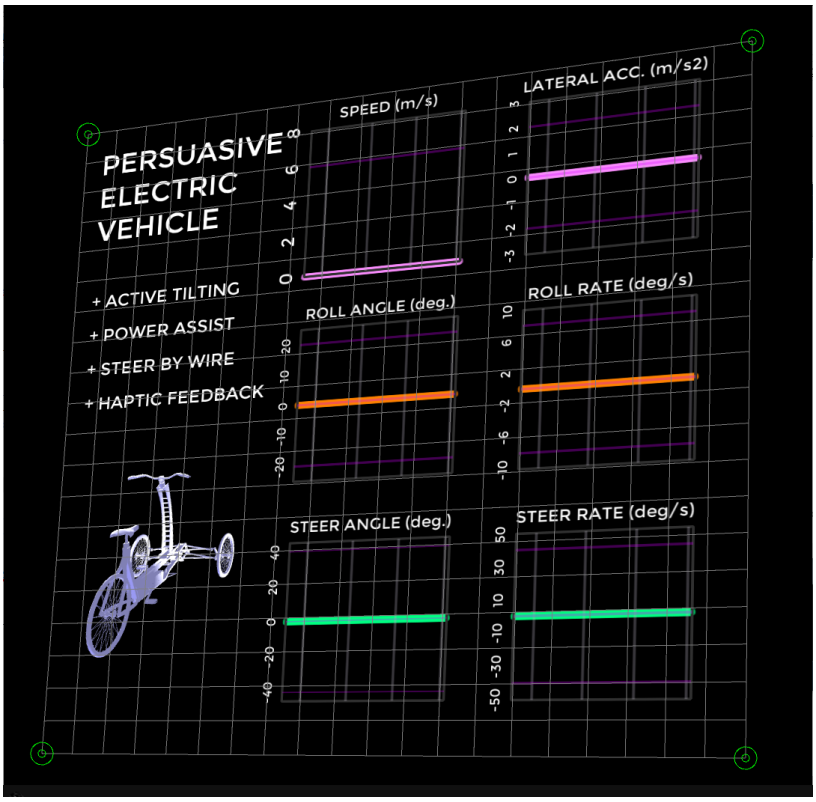


Figure 128: Live streamed data projected onto a table

Arduino

The PEV uses two Arduino MEGA boards for the acquisition of the sensor data and for the control of the motors. The reason for using two boards recalls in the VESC controller. The straight forward implementation of the VESC library for Arduino makes appropriate to use connect both through UART connection.

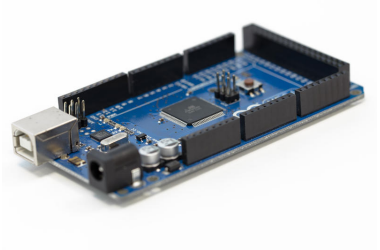


Figure 129: Arduino Mega

The Arduino Mega is able to manage 4 Serial connections simultaneously. Since the library makes use of one of them for debugging, it remains 3 Serial connections. The PEV has 4 motors incorporated (steer, handle bar, tilt, throttle), so two Arduino Megas are needed to control all the motors. The Arduinos will be connected through the I2C protocol.

The code included in both boards has been included in the appendices.

Batteries

The PEV is powered by two set of DC batteries. The rear motor requires an input voltage of 36V, which is provided by the HWT-1004-7AB battery. This battery is a removable pack, is designed as a 10S4P battery pack by using Li(NiCoMn)O2 cells and its nominal capacity is 11.4 Ah. It also meets waterproof IPX4 and provides two-level protections. The first level protection is done by software which is typically slow to act, and the second level is done by hardware which react very fast, on the order of microseconds or milliseconds. HWT-1004-7AB supplies one auxiliary power âĂŞ USB 5V, so that can satisfies the demand of variety electronical gadgets such as smart phones, MP3 player and head light.

Figure 130: HWT-1004-7AB battery

On the other side, the NIDEC motors need an input voltage of 24V. A pair of PowerSonic PS1290, each one of 12V, were selected for this task. The Power-Sonic PS-1290 is a 12 Volt 9 Amp Hour rechargeable sealed lead acid battery.

Finally, the Arduino boards are powered by the VESC controller 5V output.

Electronic Schematic

The schematic of all the components and their connections to the two Arduino Mega boards is included in the next page:

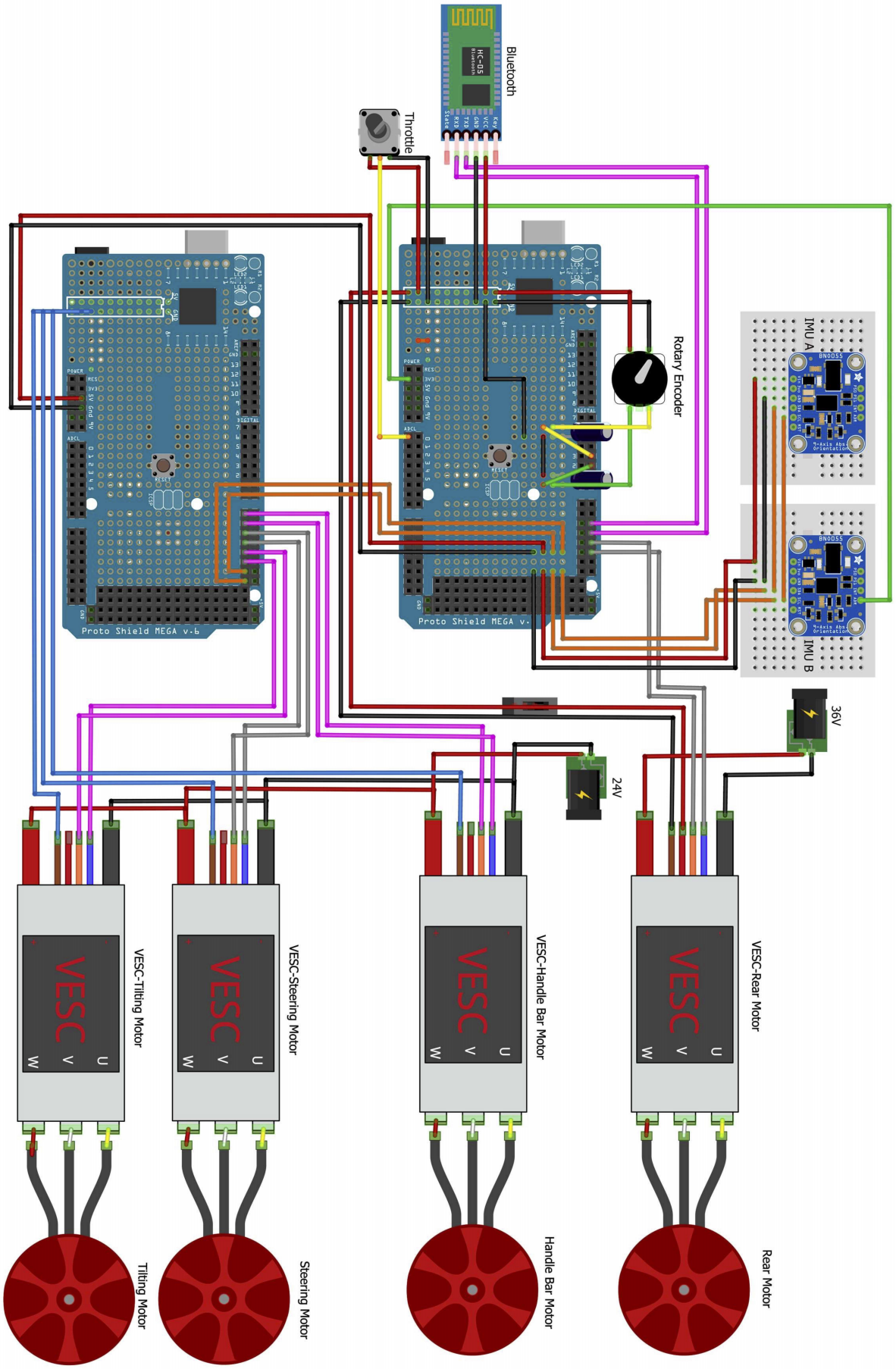


Figure 131: Electronic Schematic

6. *Vehicle and Control Tests*

In this chapter the control strategy is applied to the real scale PEV and an stability and robustness analysis is presented as well. Due to the possibility of controlling the PEV remotely (through the Bluetooth module and the Android app) two models were studied simultaneously. After the proper analysis and validation in MATLAB, the control strategy was implemented in the Arduino boards and the PEV was tested several times. Here the results of these tests are presented.

Control Strategy Analysis

The control strategy works in parallel with the dynamic model of the vehicle. It receives the sensor data as inputs; that information, along with with the gains and the actual speed of the vehicle, gives result to the output signal M_t that tilts the vehicle.

Regarding the sensors, the rotary encoder in the rear wheel and the pair of inertial motion units provide the necessary data. The longitudinal speed given by the rotary encoder is used to calculate the gains of the model as well. The actuated motor is the tilting NIDEC motor, since the steering motor only responses to the driver's inputs and the handle bar motor only does the haptic feedback to the driver.

Figure 132: Workflow of the control strategy: input from IMU and output to tilting actuator

Model Parameters

To calculate the gains of the control strategy it is first needed to obtain or measure the **parameters to complete the dynamic model**. This includes the vehicle and the wheels variables:

$$m \;\; L_f \;\; L_r \;\; I_x \;\; I_z \;\; C_f \;\; C_r \;\; \lambda_f \;\; \lambda_r \;\; ....$$

Therefore, the model presented in Chapter 3 needs to be calculated for the PEV characteristics. Both the vehicle with and without the driver were modelled in CAD software (Solidworks). By indicating the material of each body the necessary properties for the control model were extracted.

The geometrical parameters – $h \;\; L_f \;\; L_r \;\; R_{wf} \;\; R_{wr} \;\; b$ – were measured directly from the model. The dynamic parameters – $m \;\; I_x \;\; I_z \;\; ...$ –, on the other side, were extracted from the mass properties.

Name	Driver included	No driver included	**Units**	**Description**
m	115	35	kg	Total mass of the vehicle
h	0.91	0.36	m	Position of the center of gravity G on the z axis
L_f	0.718	0.518	m	Distance from center of gravity to front axle
L_r	0.61	0.81	m	Distance from center of gravity to rear axle
m_f	52.82	21.35	kg	Mass supported by the front wheels
m_r	62.18	13.65	kg	Mass supported by the rear wheel
I_x	25.14	4.00	kg m^2	Vehicle roll moment of inertia
I_z	17.8	10.93	kg m^2	Vehicle yaw moment of inertia
$I_{wheel\,f\,\theta}$	0.05	0.05	kg m^2	Tilting inertia of each front wheel about its own axis
$I_{wheel\,f\,rot}$	0.11	0.11	kg m^2	Inertia of each front wheel about its rotating axis
$I_{wheel\,f\,\phi}$	0.05	0.05	kg m^2	Yaw inertia of each front wheel about its own axis
$I_{wheel\,r\,\theta}$	0.16	0.16	kg m^2	Tilting inertia of the rear wheel about its own axis
$I_{wheel\,r\,rot}$	0.32	0.32	kg m^2	Inertia of the rear wheel about its rotating axis
$I_{wheel\,r\,\phi}$	0.16	0.16	kg m^2	Yaw inertia of the rear wheel about its own axis
$R_{w\,f}$	0.218	0.218	m	Front wheel radius
$R_{w\,r}$	0.315	0.315	m	Rear wheel radius
$F_{z\,f}$	259.10	104.71	N	Vertical reaction in each front wheel
$F_{z\,r}$	609.95	133.93	N	Vertical reaction in the rear wheel
C_f	7277.45	2690.05	N/rad	Cornering stiffness of each front wheel
C_r	20454.62	3501.36	N/rad	Cornering stiffness of the rear wheel
λ_f	231.90	79.70	N/rad	Camber stiffness of each front wheel
λ_r	731.44	105.33	N/rad	Camber stiffness of the rear wheel
b	0.92	0.92	m	Width of the vehicle in the base
M_{max}	518.949	157.941	Nm	Maximum tilt torque before rolling over

Table 7: PEV parameters extracted from Solidworks model

Driver Modelling

The driver was modeled as a 80kg person seated and in driving position. Due to the location of the body, the center of gravity and the moments of inertia changed considerably. The variability in the weight and the position of the driver will be overcome by the uncertainly analysis presented below.

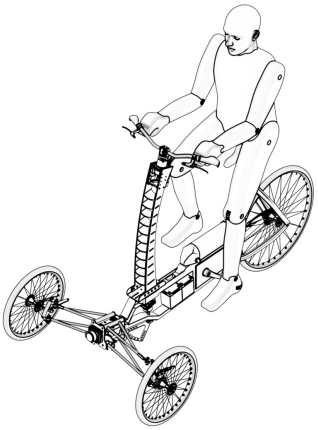


Figure 133: Driver model in the PEV

Tire Stiffness

In regards to the tires, the stability and handling depends strongly on the tire properties – cornering stiffness and camber stiffness. Mathematical models are available to predict vehicle handling. However, very little data is available on properties of bicycle and tricycle tires currently on the market.

In Windes et. al., 2013[67], the authors focus on the experimental determination of the cornering and camber stiffness of several different types of bicycle and tricycle tires. Indeed, they designed and built an machine for measuring cornering and camber stiffness using the back-to-back method. In this way, the cornering and camber stiffness were obtained over a range of vertical loads.

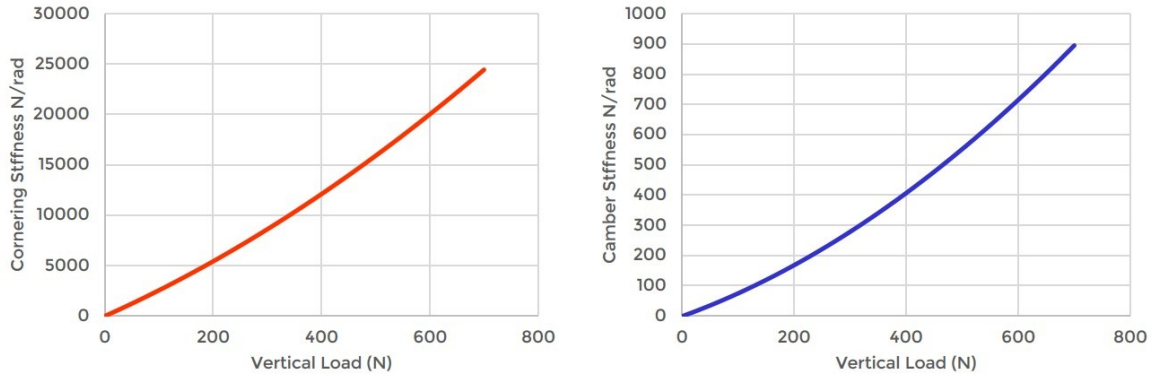
[67] Peter Windes, Mark Archibald, and Bryan Joseph. Experimental determination of bike tire stiffnesses. Proceedings of the 2013 ASEE North-Central Section Conference, 2013. Available at http://people.cst.cmich.edu/yelam1k/asee/proceedings/2013/papers/97.pdf

Knowing the mass balance of the PEV – m_f m_r – , the vertical loads of each wheel were estimated: F_{zf} F_{zr}. Then, using the quadratic relationships presented in Windes et. al., the cornering and camber stiffness were obtained.

The PEV is mounting a Continental Grand Prix 4000 SII tire, which is not modeled in the mentioned paper. Nevertheless, an average estimation from those selected tires was calculated and the relation between the stiffness and the vertical load was estimated for this tire:

$$C = \frac{F_z^2}{3690} + \frac{F_z}{2.38} \qquad \lambda = \frac{F_z^2}{66085} + \frac{F_z}{85.47}$$

Figure 134: Cornering and camber stiffness in function of the vertical load



Simulink Model

The control strategy stability and robustness was studied in MATLAB and Simulink. The model is illustrated in Figure 135.

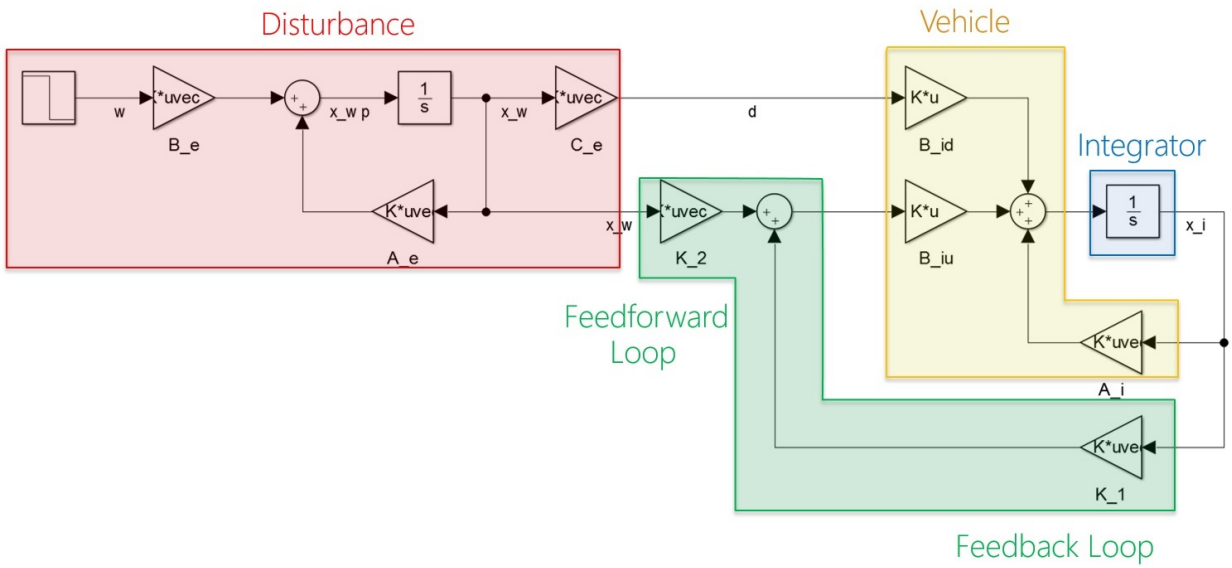


Figure 135: Simulink model

The steering input from the driver is include as a disturbance in the system, whose effect on the perceived lateral acceleration a_{per} should be canceled by the tilt torque M_t. This fact is taken into account with the feedforward gains $K_d = \begin{bmatrix} K_\delta & K_{\dot{\delta}} \end{bmatrix}^T$

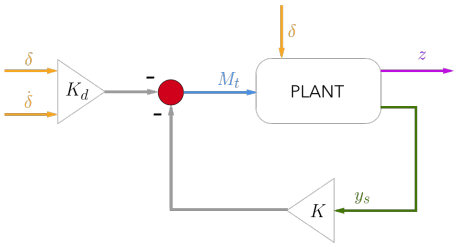


Figure 136: Control strategy schematic

The vehicle is modeled with the differential equation

$$\dot{x}_i = \begin{bmatrix} A_i \end{bmatrix} x_i + \begin{bmatrix} B_{id} \end{bmatrix} d + \begin{bmatrix} B_{iu} \end{bmatrix} u$$

This differential equation is integrated to get the state vector x_i. The regulator problem is then completed with the feedback loop of the state vector, optimized to minimize the perceived lateral acceleration a_{per}.

For the next sections, it is fundamental to do a distinction between the open and the closed loop systems:

–Open-loop system

$$\dot{x}_s = \begin{bmatrix} A_s \end{bmatrix} x_s + \begin{bmatrix} B_s \end{bmatrix} u$$
$$y = \begin{bmatrix} I \end{bmatrix}_{7x7} x_s$$

–Closed-loop system

$$\dot{x}_s = \Big(\begin{bmatrix} A_s \end{bmatrix} - \begin{bmatrix} B_s \end{bmatrix} \begin{bmatrix} K \end{bmatrix} \Big) x_s + \begin{bmatrix} B_s \end{bmatrix} u$$
$$y = \begin{bmatrix} I \end{bmatrix}_{7x7} x_s$$

with $x_s = \begin{bmatrix} \dot{y} & \dot{\psi} & \theta & \dot{\theta} & a_{per} & \delta & \dot{\delta} \end{bmatrix}^T$

Feedback and Feedforward Gains

Following the methodology explained in *Chapter 3 - Control Strategy Summary*, the feedback and feedforward gains are obtained. The influence of the inverse velocity term in the gain scheduling stage was also studied.

Once the dynamic model is completed with the PEV parameters, the gains are obtained from the Riccati and Sylvester equations:

- Feedback Gains: Riccati equation

$$M_1 A_i + A_i^T M_1 - M_1 B_{iu} R_u^{-1} B_{iu}^T M_1 + Q_x = 0$$

$$K = R_u^{-1} B_{iu}^T M_1$$

- Feedback Gains: Slyvester equation

$$M_2 A_e + (A_i^T - M_1 B_{iu} R_u^{-1} B_{iu}^T) M_2 + M_1 B_{id} C_e = 0$$

$$K_d = R_u^{-1} B_{iu}^T M_2$$

The obtained gains are transformed so that the state vector is fully measurable:

$$K'_{a_{per}} = \frac{K_{\dot{y}}}{a_{11} + h a_{41} + K_{\dot{y}} (b_{u1} + h b_{u4})} \tag{56}$$

$$K'_{\dot{\psi}} = \frac{K_{\dot{\psi}}(a_{11} + h a_{41}) - K_{\dot{y}}(a_{12} + h a_{42} + V_x)}{a_{11} + h a_{41} + K_{\dot{y}} (b_{u1} + h b_{u4})} \tag{57}$$

$$K'_{\theta} = \frac{K_{\theta}(a_{11} + h a_{41}) - K_{\dot{y}}(a_{13} + h a_{43} - g)}{a_{11} + h a_{41} + K_{\dot{y}} (b_{u1} + h b_{u4})} \tag{58}$$

$$K'_{\dot{\theta}} = \frac{K_{\dot{\theta}}(a_{11} + h a_{41}) - K_{\dot{y}}(a_{14} + h a_{44})}{a_{11} + h a_{41} + K_{\dot{y}} (b_{u1} + h b_{u4})} \tag{59}$$

$$K'_{a_{per}^I} = \frac{K_{a_{per}^I}(a_{11} + h a_{41})}{a_{11} + h a_{41} + K_{\dot{y}} (b_{u1} + h b_{u4})} \tag{60}$$

$$K'_{\delta} = \frac{K_{\delta}(a_{11} + h a_{41}) - K_{\dot{y}}(b_{\delta 1} + h b_{\delta 4})}{a_{11} + h a_{41} + K_{\dot{y}} (b_{u1} + h b_{u4})} \tag{61}$$

$$K'_{\dot{\delta}} = \frac{K_{\delta}(a_{11} + h a_{41})}{a_{11} + h a_{41} + K_{\dot{y}} (b_{u1} + h b_{u4})} \tag{62}$$

$$K' = \begin{bmatrix} K'_{a_{per}} & K'_{\dot{\psi}} & K'_{\theta} & K'_{\dot{\theta}} & K'_{a_{per}^I} & K'_{\delta} & K'_{\dot{\delta}} \end{bmatrix}$$

	K_C	K_V	$K_{1/V}$
$K'_{a_{per}}$	-0.56	-0.27	0.52
$K'_{\dot{\psi}}$	19.14	-4.09	-16.63
K'_{θ}	242.92	-2.51	5.54
$K'_{\dot{\theta}}$	64.40	-0.46	0.84
$K'_{a_{per}^I}$	-0.32	0	0
K'_{δ}	1228.56	-270.84	-1121.79
$K'_{\dot{\delta}}$	220.54	-59.70	-201.26

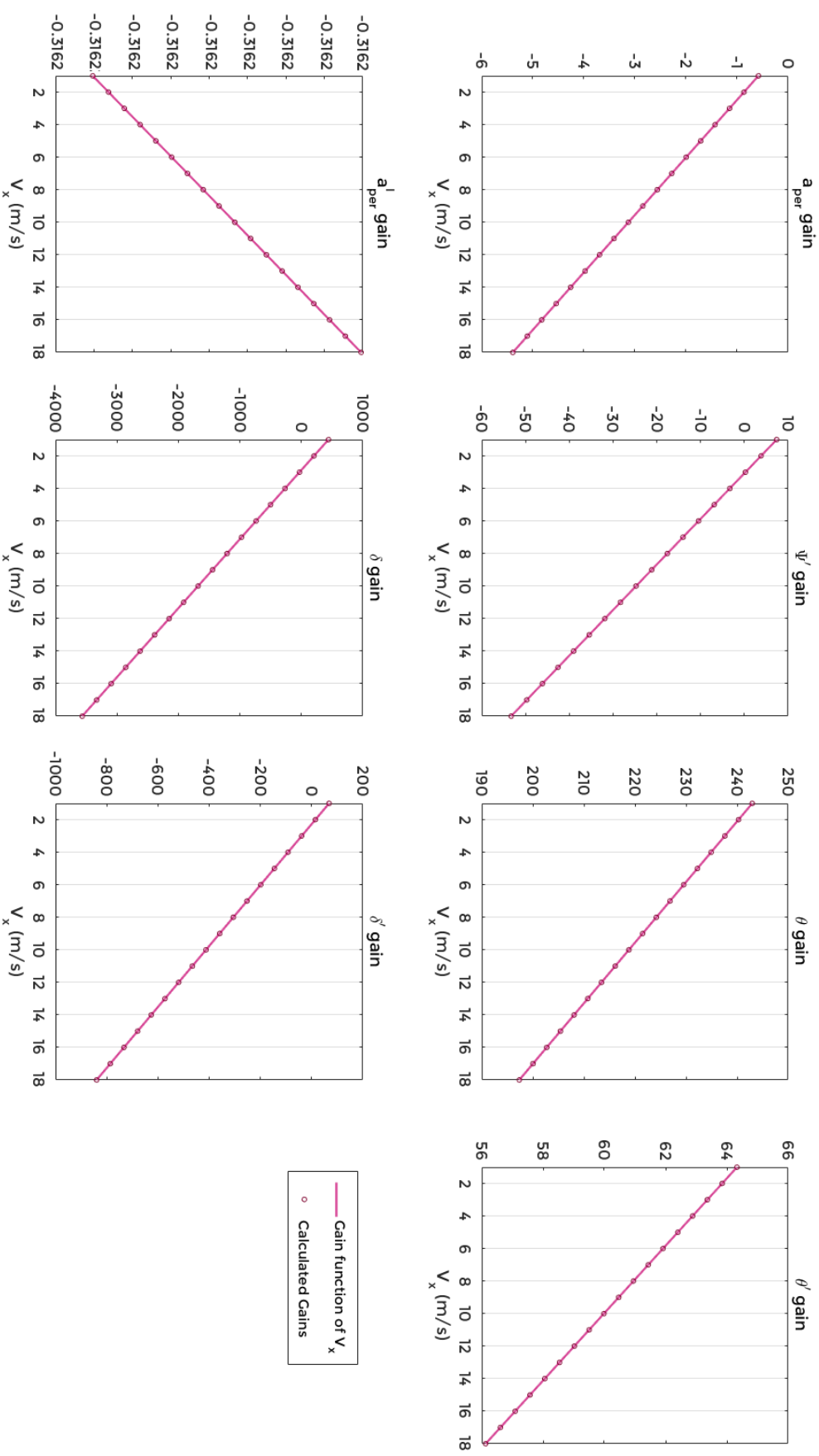


Figure 137: Gains relation with velocity. Velocity inverse term is not considered

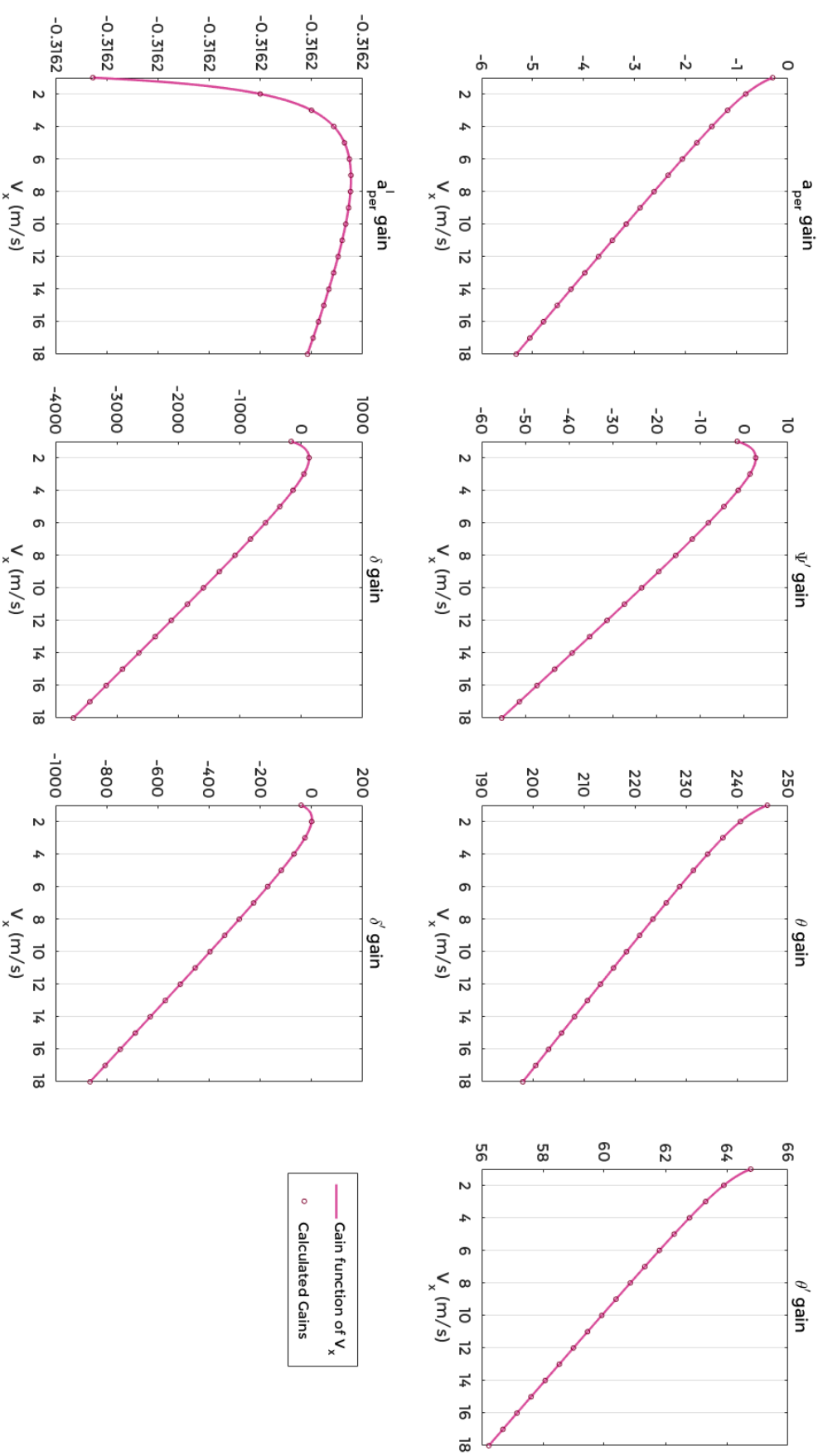


Figure 138: Gains relation with velocity. Velocity inverse term is considered

In Figures 137 and 138 the gains of the PEV without driver have been represented. The gains curves are estimated as a least square estimation of some discrete points. As was indicated in previous chapters, the gain scheduling as function of the longitudinal velocity was designed to depend on three terms K_c, K_V and $K_{1/V}$ with:

$$\begin{pmatrix} K_c \\ K_V \\ K_{1/V} \end{pmatrix} = (M^T M)^{-1} M^T K_M$$

If the inverse term $K_{1/V}$ is not considered, then the gain scheduling only depends linearly on the velocity:

$$\begin{pmatrix} K_c \\ K_V \end{pmatrix} = (M^T M)^{-1} M^T K_M$$

Stability

If the system is studied without feedback control, it is inherently unstable. The instability can be visualized through different indicators.

The system matrix $[A_s]$ is non-negative defined, meaning that it has positive eigenvalues. The eigenvalues of $[A_s]$ represent the open poles of the system, and if a pole is in the positive side of the real numbers, then it means that the system is unstable.

$$eig([A_s]) = \begin{bmatrix} 0 \\ -92.47 \\ -43.38 \\ -4.06 \\ 3.61 \\ -0.5 \\ -1 \end{bmatrix}$$

The eigenvalues are usually represented in the pole-zero map. The poles (represented with a cross *X*) are the roots of denominator of the system transfer function, while the zeros (represented with a circle *O*) are the roots of the numerator. In figure 139, a pole-zero map has been included for each system transfer function. The input is the torque M_t and the outputs are the remaining systems variables: $a_{per} \quad \dot{\psi} \quad \theta \quad \dot{\theta} \quad per^I$.

All transfer functions share a common pole in the positive real axis, which makes the system unstable. The transfer functions can also share some poles and zeros.

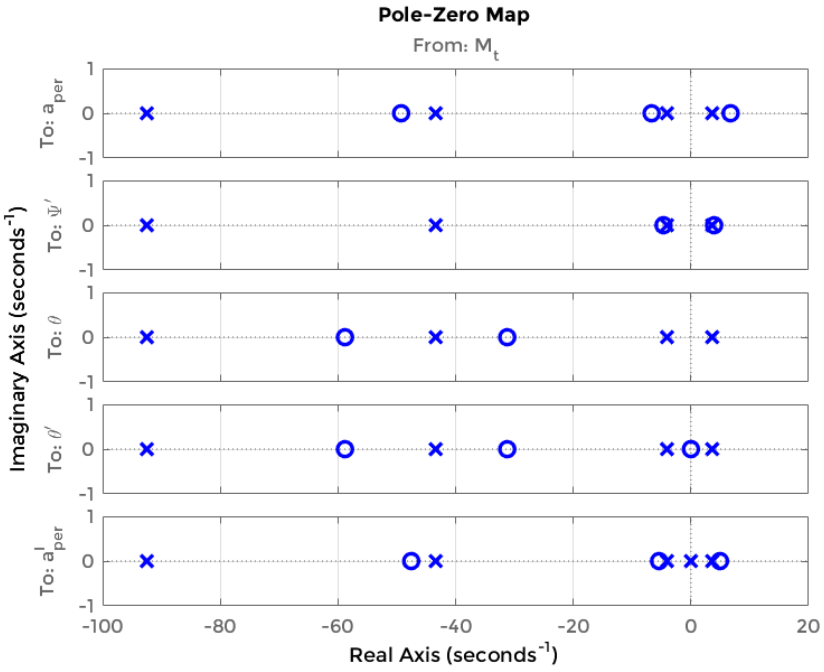


Figure 139: Pole zero map of the open–loop system (unstable)

Simulating the response of the system to a disturbance δ, $\dot{\delta}$ makes the state variables unstable and tend to infinite values:

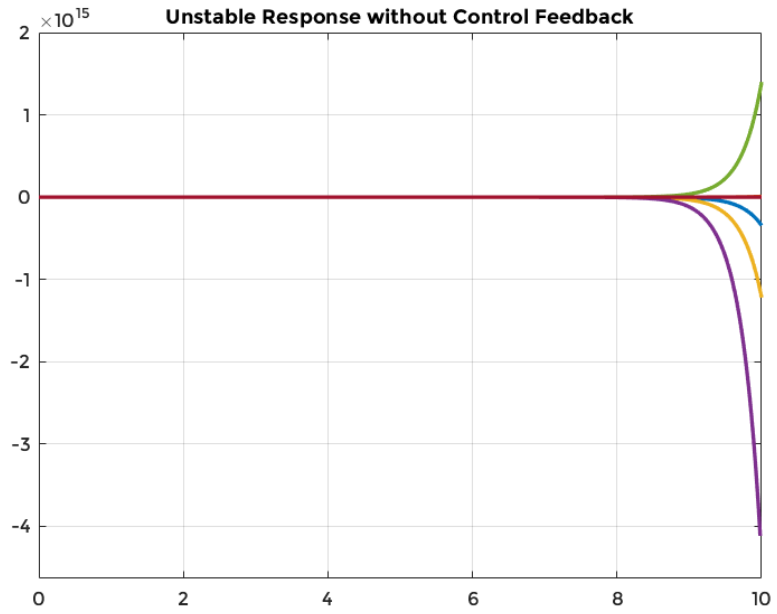


Figure 140: Unstable response of the open–loop system, without control feedback

Looking at the ranks of the observability and controllability matrix of the system:

$$Ob = \begin{bmatrix} C & C\,A & C\,A^2 & \dots & C\,A^6 \end{bmatrix}^T$$

$$Co = \begin{bmatrix} B & A\,B & A^2\,B & \dots & A^6\,B \end{bmatrix}$$

The system is observable if Ob has full rank (A_{7x7}) and is controllable if Co has also full rank (A_{7x7}). Therefore, the open loop system is observable ($rank(Ob) = 7$), but not controllable ($rank(Ob) = 7$)

If the system is controlled with the feedforward and feedback gains, then the system stabilizes. The matrix $[A_s]$ is definite non-positive, with all its eigenvalues located in the negative side of the pole map. However, a pair of conjugate imaginary poles and positive zeros are introduced to effectively stabilize the system.

$$eig([A_s] - [B_s][K]) = \begin{bmatrix} -81.87 \\ -44.56 + 11.53i \\ -44.56 - 11.53i \\ -4.917 \\ -5.478 \\ -0.50 \\ -1 \end{bmatrix}$$

The pole-zero map and the response of the system are indeed stable:

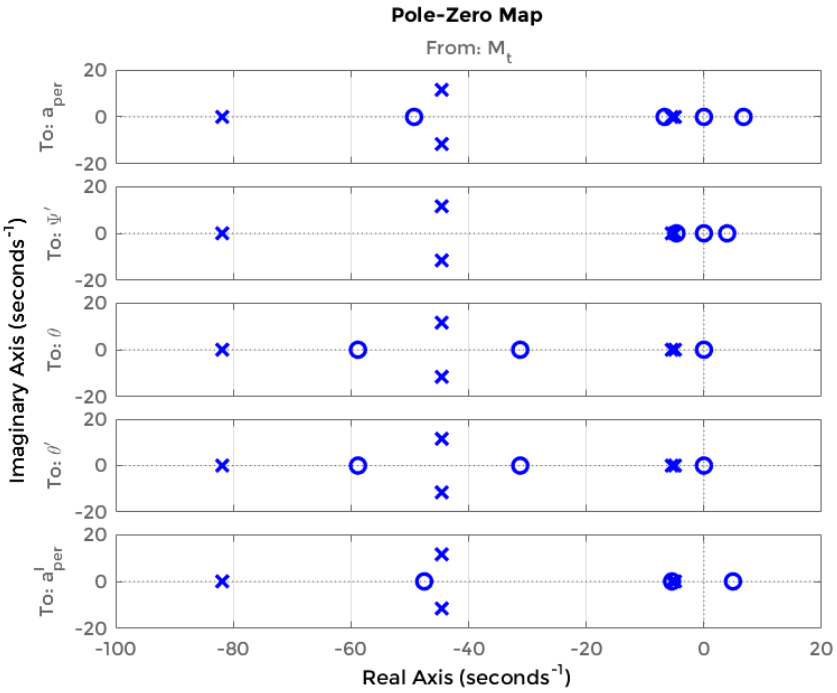


Figure 141: Pole zero map of the closed–loop system (stable)

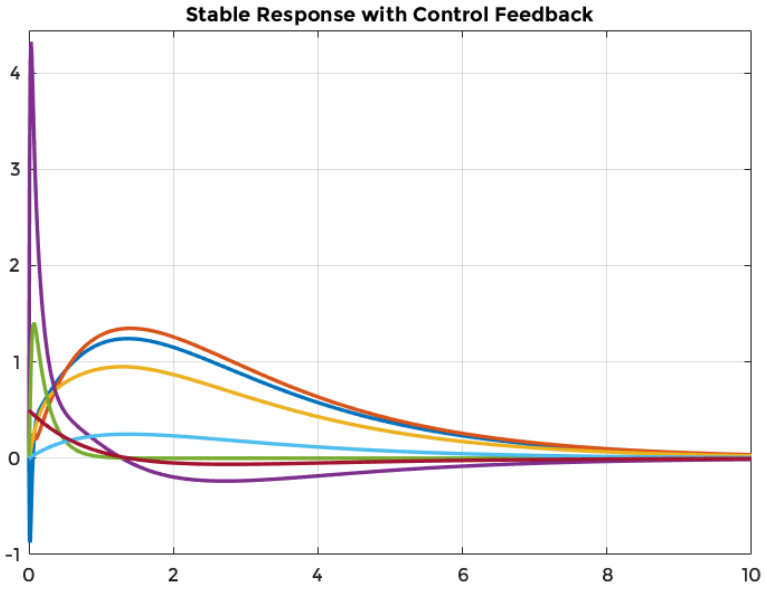


Figure 142: Stable response of the closed–loop system, with control feedback

Robustness

The μ-analysis makes it possible to carry out a posteriori robust studies with respect to the parametric variations of the model, or the neglected dynamics, taking into account the structure and the nature of the uncertainties.

According to the generalized small gains theorem, checking the robustness of the stability of the closed loop, amounts to calculating the singular values of the transfer function $\mu(H(jw))$. The system is stable if these values are under 1 $\mu(H(jw)) < 1$ and if in addition $\mu(H(jw))$ is small, the stability of the system is robust.

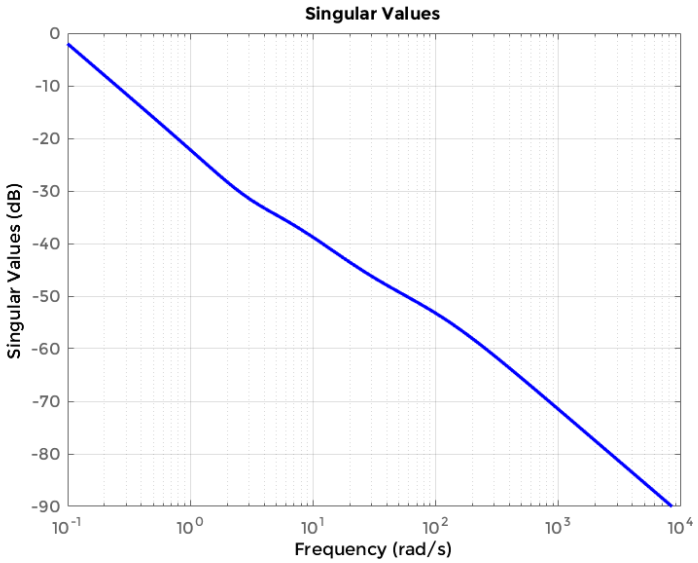


Figure 143: Singular values of the open–loop system

High-gain feedback in low-frequency ranges is a way to deal with the effects of unknown biases and disturbances acting on the process output. The control feedback increases the singular values (the principal gains of the frequency response) at 10 to 100 Hz range, but it remains below $\mu = 0\,dB = 20\log(1)$.

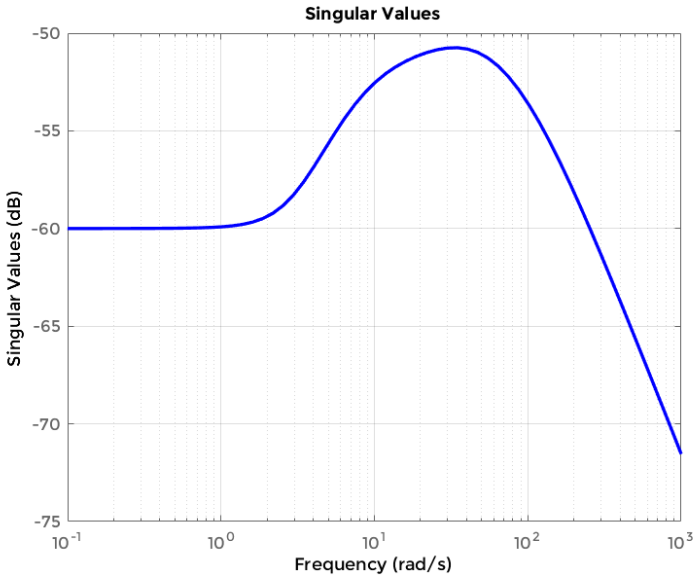


Figure 144: Singular values of the closed–loop system

Apart from being stable, the system has deal with the effects of uncertainty. Reducing the effects of some forms of uncertainty (initial conditions, low-frequency disturbances) without increasing the effects of the sensor noise or the model uncertainty is the primary job of the feedback control system.

At the heart of robust control is the concept of an uncertain LTI system. Model uncertainty arises when system gains or other parameters are not precisely known, or can vary over a given range. The following table summarizes the model parameters and their variation over the nominal value.

	Value	**Deviation%**	
L_f	0.51	5	m
L_r	0.81	5	m
h	0.36	15	m
m	35	20	kg
I_z	11	50	$kg\,m^2$
I_x	4	50	$kg\,m^2$
C_f	3500	25	N/rad
C_r	3000	25	N/rad
λ_f	200	25	N/rad
λ_r	200	25	N/rad

Table 8: Parameters Deviations

Once formulated, high-level system robustness tools can help to analyze the potential degradation of stability and performance of the closed-loop system brought on by the system model uncertainty.

The parameter uncertainties mean uncertain pole and zero locations. Fortunately, among the represented poles, none of them is moving towards the positive real part, which will produce instabilities.

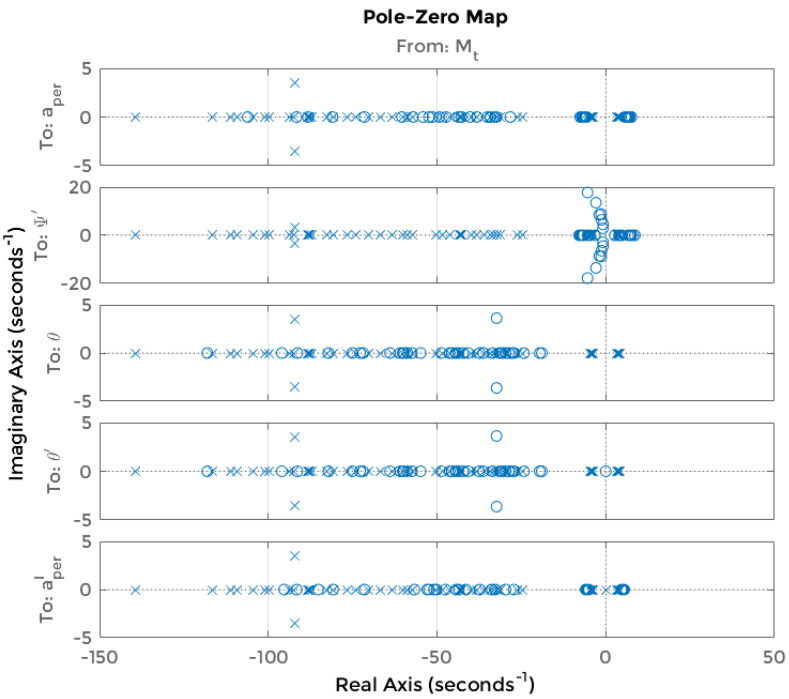


Figure 145: Uncertainty in the pole zero map of the open–loop system

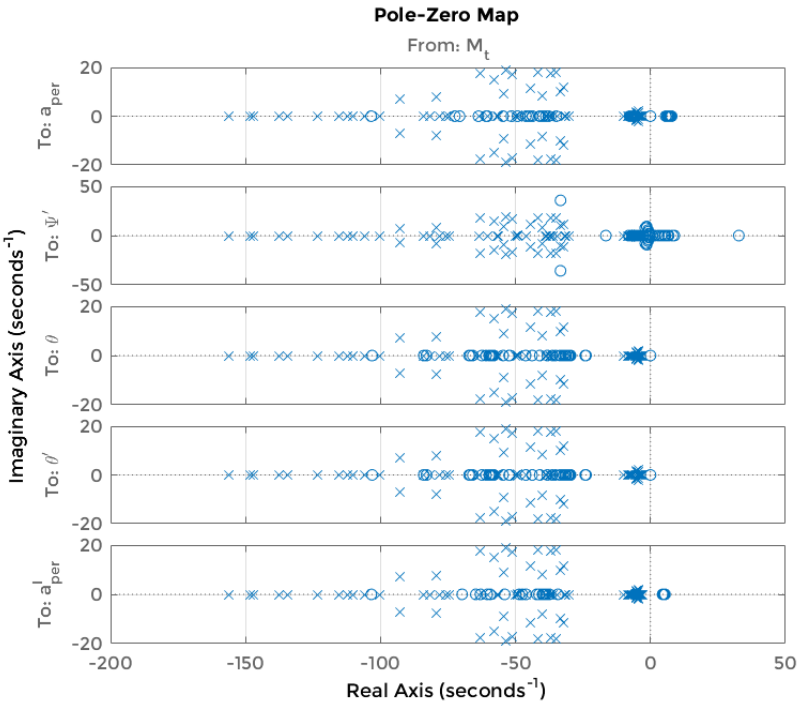


Figure 146: Uncertainty in the pole zero map of the closed–loop system

Regarding the singular values of the closed-loop system, these values are not affected by the parameters uncertainties.

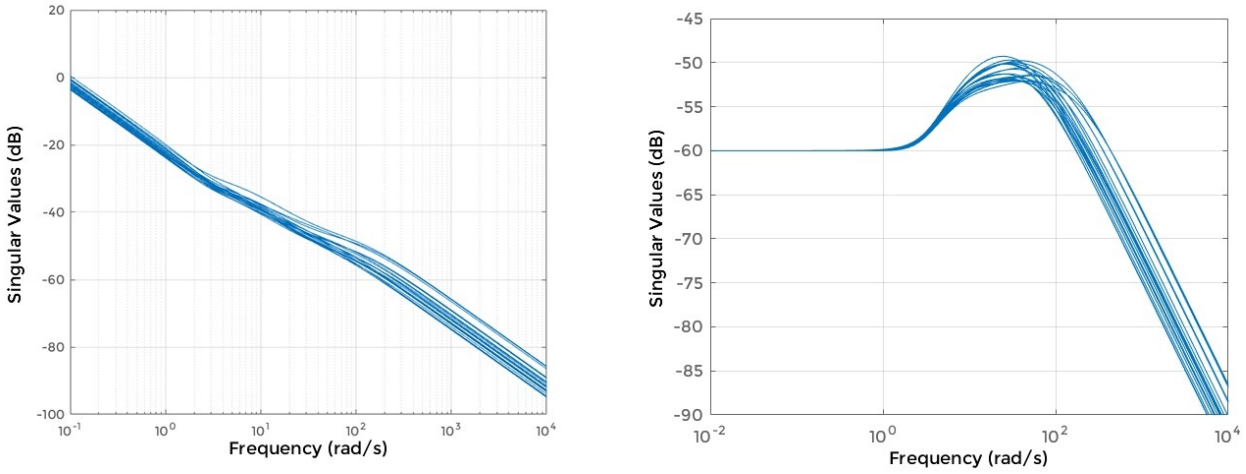


Figure 147: Uncertainty in the singular values of the open and closed–loop system

Notions such as gain and phase margins help to quantify the sensitivity of stability and performance in the face of model uncertainty, which is the imprecise knowledge of how the control input directly affects the feedback variables.

The gain margin is the amount of gain increase or decrease required to make the loop gain unity at the frequency where the phase angle is 180°. Similarly, the phase margin is the difference between the phase of the response and 180° when the loop gain is 1.

```
GainMargin: 0.0707
GMFrequency: 4.4670
PhaseMargin: 76.0715
PMFrequency: 50.6347
DelayMargin: 0.0262
DMFrequency: 50.6347
```

Figure 148: Gain and phase margin

If the stability robustness margin is greater than 1 it means that the uncertain system is stable for all values of its modeled uncertainty. Being less than 1 implies that certain allowable values of the uncertain elements lead to instability. In our system only bounds on the exact stability margin were computed. The exact robust stability margin is guaranteed to lie in between these upper and lower bounds.

```
LowerBound: 0.9099
UpperBound: 1.9902
DestabilizingFrequency: 0.1005
```

A nominally stable uncertain system is generally unstable for specific values of its uncertain elements. Determining the values of the uncertain elements closest to their nominal values for which instability occurs is a robust stability calculation.

```
Destabilizing Parameters:
        C_f: 5.2414e+03
        C_r: 1.5074e+03
        I_x: 0.0196
        I_z: 0.0540
        L_f: 0.5607
        L_r: 0.7294
        h: 0.2525
        landa_f: 299.5095
        landa_r: 100.4905
        m: 48.9313
```

If the uncertain system is stable for all values of uncertain elements within their allowable ranges, the uncertain system is robustly stable. Conversely, if there is a combination of element values that cause instability, and all lie within their allowable ranges, then the uncertain system is not robustly stable.

```
Uncertain system is possibly not robustly stable to modeled uncertainty.
-- It can tolerate up to 91% of the modeled uncertainty.
-- A destabilizing combination of 199% of the modeled uncertainty was found.
-- This combination causes an instability at 0.1 rad/seconds.
-- Sensitivity with respect to the uncertain elements are:
'C_f' is 27%. Increasing 'C_f' by 25% leads to a 7% decrease in the margin.
'C_r' is 17%. Increasing 'C_r' by 25% leads to a 4% decrease in the margin.
'I_x' is 78%. Increasing 'I_x' by 25% leads to a 20% decrease in the margin.
'I_z' is 28%. Increasing 'I_z' by 25% leads to a 7% decrease in the margin.
'L_f' is 6%.  Increasing 'L_f' by 25% leads to a 2% decrease in the margin.
'L_r' is 9%.  Increasing 'L_r' by 25% leads to a 2% decrease in the margin.
'h' is 8%.    Increasing 'h' by 25% leads to a 2% decrease in the margin.
'landa_f' is 5%. Increasing 'landa_f' by 25% leads to a 1% decrease in the margin.
'landa_r' is 5%. Increasing 'landa_r' by 25% leads to a 1% decrease in the margin.
'm' is 12%. Increasing 'm' by 25% leads to a 3% decrease in the margin.
```

Experiments

Once the control system was developed and the PEV was fully fabricated, the test and validation stage started. First, some tests were run to validate the sensor data. Second, the electronics were tested, along with the live streaming of data through Bluetooth. Finally, the control feedback was implemented in Arduino and tested several times.

Figure 149: PEV outside the MIT Media Lab

Test Setup

With the intention of comparing the experimental results with the carried out simulations, the PEV was tested under some controlled conditions. The 6th floor of the MIT Media Lab was selected as the test field due to the available space and the accessibility of the room. The geometry of the experiment was limited by two circumferences of known radius ($R = 2.5m$). These circumferences were delimited by some cardboard boxes laid on the floor.

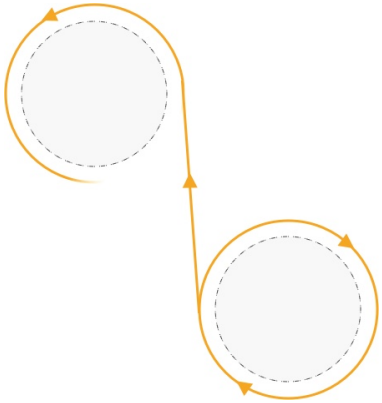


Figure 150: Testing setup schematic

As has been stated below, the first tests were run to validate the data coming from the sensors. The throttle potentiometer did not implicate any problem, neither did the rotary encoder at the rear wheel. On the contrary, the pair of IMUs (in the frame and in the handle bar) presented some problems.

- In the calibration stage, the IMUs seemed to lose the 'north pole' every time the PEV was transported from one floor to another. However, when testing in the same floor, these problem did not appear. To solve this issue, the IMUs had to be calibrated again with each test.

- This calibration problem also affected considerably the initial absolute orientation of both IMUs, meaning that the relative angle between them –for the δ angle, for example– had a initial drift.

- The noise coming from the accelerometer and the gyroscope was excessive for a good control feedback. Each signal coming from the IMU was filtered with a simple one dimensional Kalman filter. Some preliminary tests were necessary to estimate the parameters of each Kalman filter: p, q, k.

The data was gathered using one of the Bluetooth modules, that sent the data to the computer, where the selected variables were saved in a .csv file. Afterwards, the data and the video from the front camera were a synchronized using an script in Matlab.

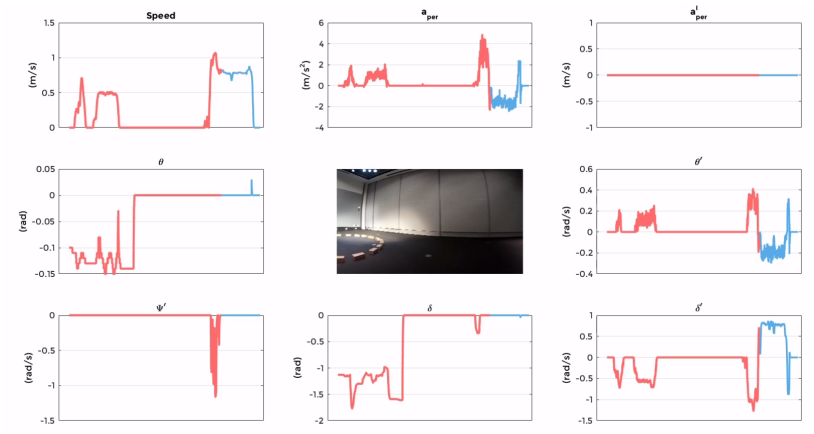
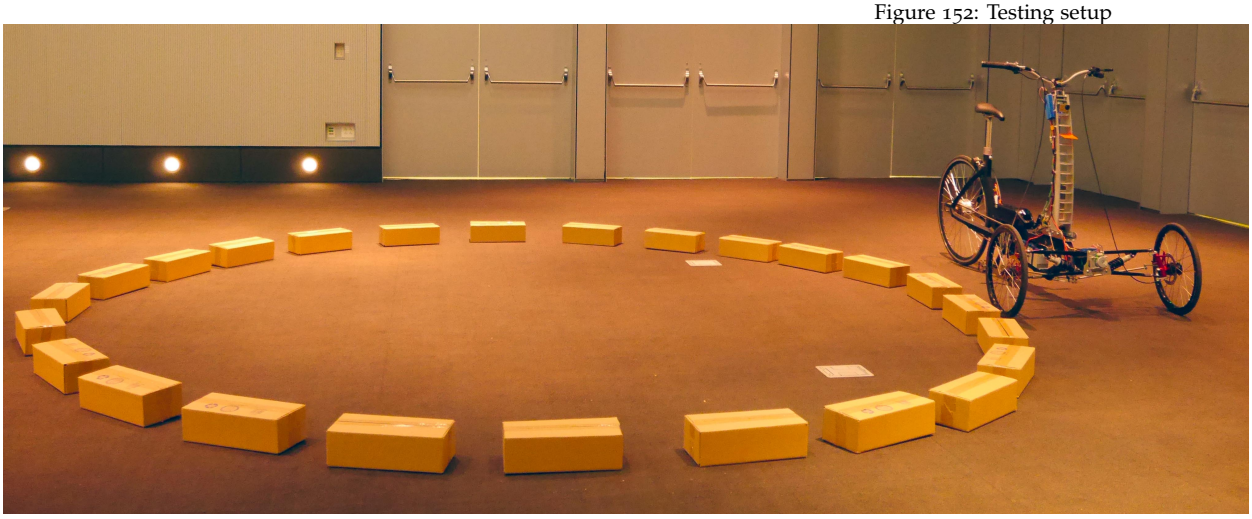


Figure 151: Live streaming through Bluetooth module

The PEV was tested with and without the tilting mechanism, thanks to the modularity of the front suspension design. The non-tilting PEV was used for the tests where the sensor data was being validated and for the tests where the control strategy was implemented. After all the systems had been validated, the PEV was modified to allow the tilting and the final experiments were carried out.

Figure 152: Testing setup



Test Results

The improvement of the PEV from the first to the last test was evident. The vehicle was exhaustively tested to improve the steering response as well as the remote control from the Android app. The mechanical robustness of the vehicle was put to the test during these experiments.

In the next set of pictures (Figure 153), two turns to the circumference are illustrated. Before running any test, the PEV team consensually decided not to put in risk the integrity of any member, so it was decided to experiment only with driverless PEV. The main focus of these tests was the validation of the mechanics, the electronics and the control strategy.

It is important to recall that at the test stage any false step can imply the breaking of the PEV or in the worst case scenario, the harm of the researches. A testing vehicle can suppose a danger if it gets out of control, so some safety measures, both at the hardware and the software level were implemented.

Therefore, since it was a driverless test, the PEV was remotely controlled, both at the steering and at the throttle. In this experiment, the control of the PEV was smooth, and the data was correctly gathered for the afterwards analysis.

With regards to the tilting mechanism, it successfully worked properly, meaning that the front suspension geometry was appropriate and that the selection of the components was satisfactory. In addition, the self limitation of the geometry limited the tilting angle to a maximum, thus preventing the PEV from rolling over when tilting.

Figure 153: Testing scene

Even though the control strategy was not fully validated, the implementation of the feedback and feedforward gains showed that the torque M_t was giving good inputs for the tilting.

Figure 154: PEV tilting during the test

The main problem with the control strategy was focused on the integration of the perceived lateral acceleration (a^I_{per}). The integration of a signal coming from the IMU was reviewed in the chapter 4, where the velocity of the MiniPEV was estimated from the longitudinal linear acceleration. The similarities between the two challenges seem obvious. That is why the control strategy had to be finally reduced to the output variable a_{per} instead of a^I_{per}

In figures 155,156, 158 and 157 a sample of the carried out test has been included. This experiment was carried out allowing the tilting of the vehicle, and the control strategy was almost similar to the one presented previously in this chapter. The only difference can be found in the output variable, that is the a_{per} instead of a^I_{per}.

As was already mentioned, lateral stability is obtained when $a_{per} = 0$. The control of the integrated value of a_{per} avoids the model errors due to parameters uncertainty, neglected dynamics or linearization of the model. Therefore, these are the errors that can appear if only the a_{per} is controlled. In this case the process to obtain the gains is completely similar.

The Figure 155 represents the disturbance of the model, that is, the input from the driver. The steering angle indicates the desired direction. In this case, the vehicle does turn to the left before turning to the right. The reader can visualize the inherent noise in the steering rate $\dot{\delta}$. Even though the Kalman filter revokes some of the signal noise, it is important to remind that the gyroscope data is very prone to vibrations.

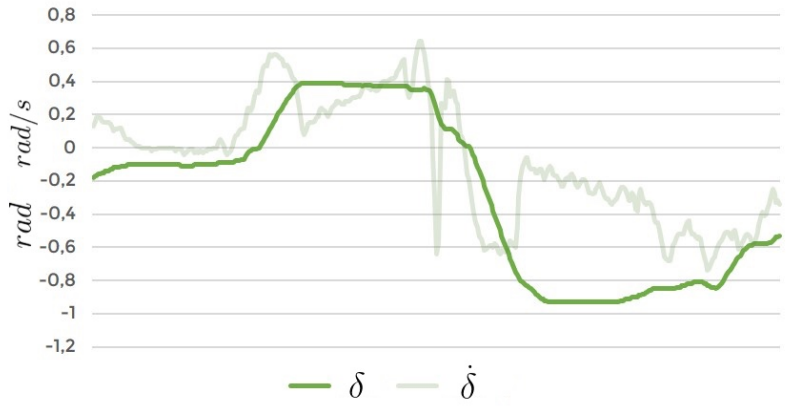


Figure 155: Disturbance signal; input from the driver δ and $\dot{\delta}$

The velocity of the vehicle has been included in Figure 156. This test was carried out at low speed, in order to have the vehicle under control if necessary. The lateral perceived acceleration a_{per} is noisy as well, whereas the yaw rate $\dot{\psi}$ is more smooth. Anyway, the data confirms that both variables are properly calculated.

Before any input from the driver –when the vehicle is going in a straight line– both the a_{per} and the $\dot{\psi}$ are null. Just when the driver turns the handle bar and steers the front wheels, the yaw rate starts to increase (meaning a left turn) and the perceived acceleration takes negative values synchronously. Just when the steering angle is changed to the opposite direction, these two signals invert their values.

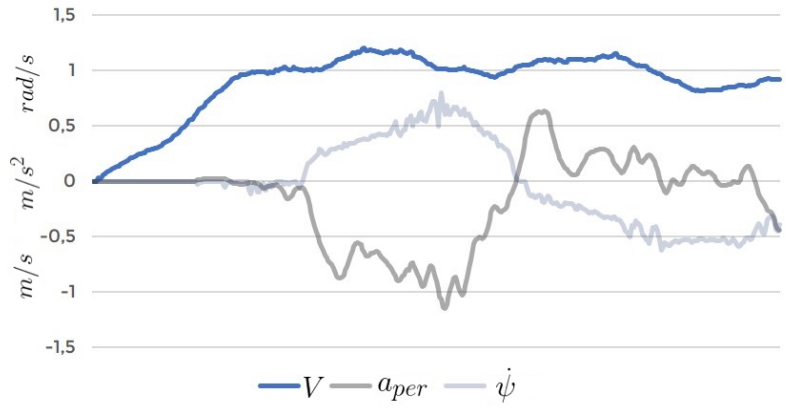


Figure 156: Vehicle variables: longitudinal velocity, perceived lateral acceleration and yaw rate

The response of the control strategy is to apply a torque M_t so that the effect of the disturbance δ on the perceived lateral acceleration a_{per} is canceled. Using the gains calculated previously, the torque required from the tilting motor is represented in the Figure 157.

The control objective is obtained with very satisfying performances; a_{per} does not exceed $1 m/s^2$ and the maximum tilting torque value is 10 Nm. The fact that more torque is required to the right turn is due to the fact that the steering input is higher in this case, so that demanding more torque.

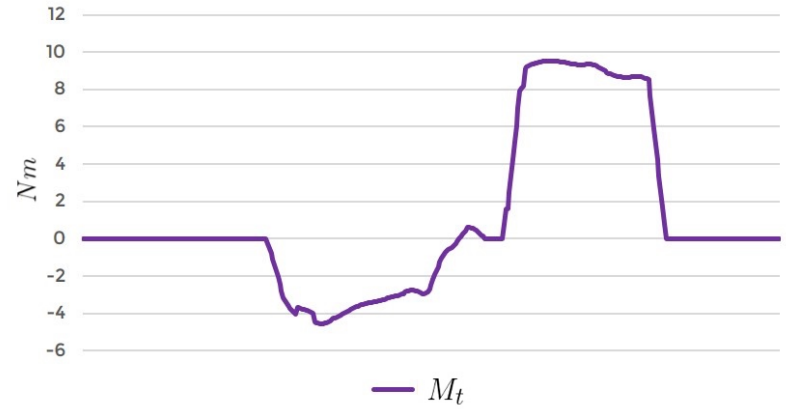


Figure 157: Torque required by the control system to tilt the PEV

The applied torque in the tilting motor leans the PEV and as a consequence, there is a change in the tilting angle θ and in the tilting rate $\dot{\theta}$. In this case the maximum tilting angle goes over $0.15rad = 8deg.$, which is not very high due to the low speed of the test.

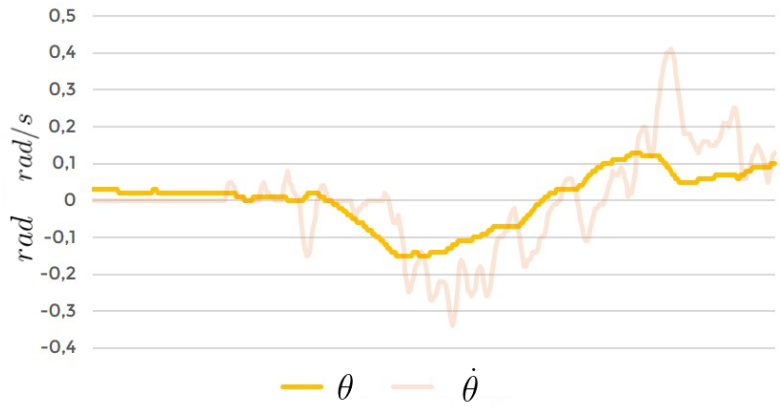


Figure 158: Control strategy results

Overall, the carried out tests were satisfactory in almost every way. The exclusion of the integrate of the perceived lateral acceleration from the control strategy changed the properties and the robustness of the system, but far from ruining the developed control model, this fact introduced a much simpler and effective model. While it is true that some errors can rise –parameters uncertainty, neglected dynamics or model linearization – the reduced model successfully tilt the vehicle.

Figure 159: PEV testing

7. Project Cost & Overview

Cost

This section covers the cost of the elements needed for the prototypes built in this thesis. It takes into account the materials, machining processes and labor costs.

Materials Costs

The material costs are obtained from the suppliers where the parts were purchased. Overall the project was carried out using already available parts from Changing Places group or MIT Media Lab Machine Shop.

Item	Site	Qty	Price	Total	Currency
Adafruit 9-DOF IMU Fusion Breakout - BNO055	Amazon	1	32,33	32,33	$
UNO R3 Board ATmega328P ATMEGA16U2	Amazon	1	10,86	10,86	$
18-8 Stainless Steel Male-Female Hex Thread Adapter	McMaster	2	10,13	20,26	$
ICAN Carbon Fiber Seat Clamp 31.81mm	Amazon	1	9,99	9,99	$
Cycling Freewheel Threaded 34 mm	Amazon	1	3,54	3,54	$
3 Speed 16-19-22T Thread Freewheel	Amazon	1	14,99	14,99	$
Threaded Steel 9 Speed 13-32T Freewheel	Amazon	1	15,99	15,99	$
Super-Swivel Ball Joint Rod Ends	McMaster	4	9,92	39,68	$
USB-C-HDMI Adapter	Amazon	1	15,99	15,99	$
Rod-End Washer Insert	McMaster	8	3,31	26,48	$
Rotary Motion Position-Measuring Transmitter	McMaster	1	307,42	307,42	$
Polyurethane Wheel for Transmitter Kit	McMaster	1	54	54	$
Signswise 68*118mm Vp-bc73 Bottom Bracket	Amazon	1	14	14	$
				565,53	$

Table 9: Material Costs

Machining Costs

The two prototypes have been taken into account (the miniPEV and the full scale PEV) throughout all the used processes:

Prototype/Process	Waterjet	Laser Cut	3D Printing	Bending	Sand Blaster	Tap	Drilling	
miniPEV		12	6	2				
PEV	15		15		10	5	6	
Total Hours	15	12	21	2	10	5	6	
Cost ($/h)	150	40	80	20	30	10	30	
Total Cost	2250	480	1680	40	300	50	180	4980

Table 10: Machining Costs

Labor Costs

Labor	Unit price ($/h)	Qty.	Total price ($)
Research Assistant	12	1400	16800
Principal Investigator	85	10	850
Total			17650

Table 11: Labor Costs

The labor costs are estimated from the annual salaries of the involved members and the machining costs are estimated from Ulrich & Eppinger's Product Design and Development [68].

[68] K.T. Ulrich and S.D. Eppinger. *Product Design and Development*. McGraw-Hill/Irwin series in marketing. McGraw-Hill/Irwin, 2004. Available at https://books.google.co.uz/books?id=Qju5AAAAIAAJ

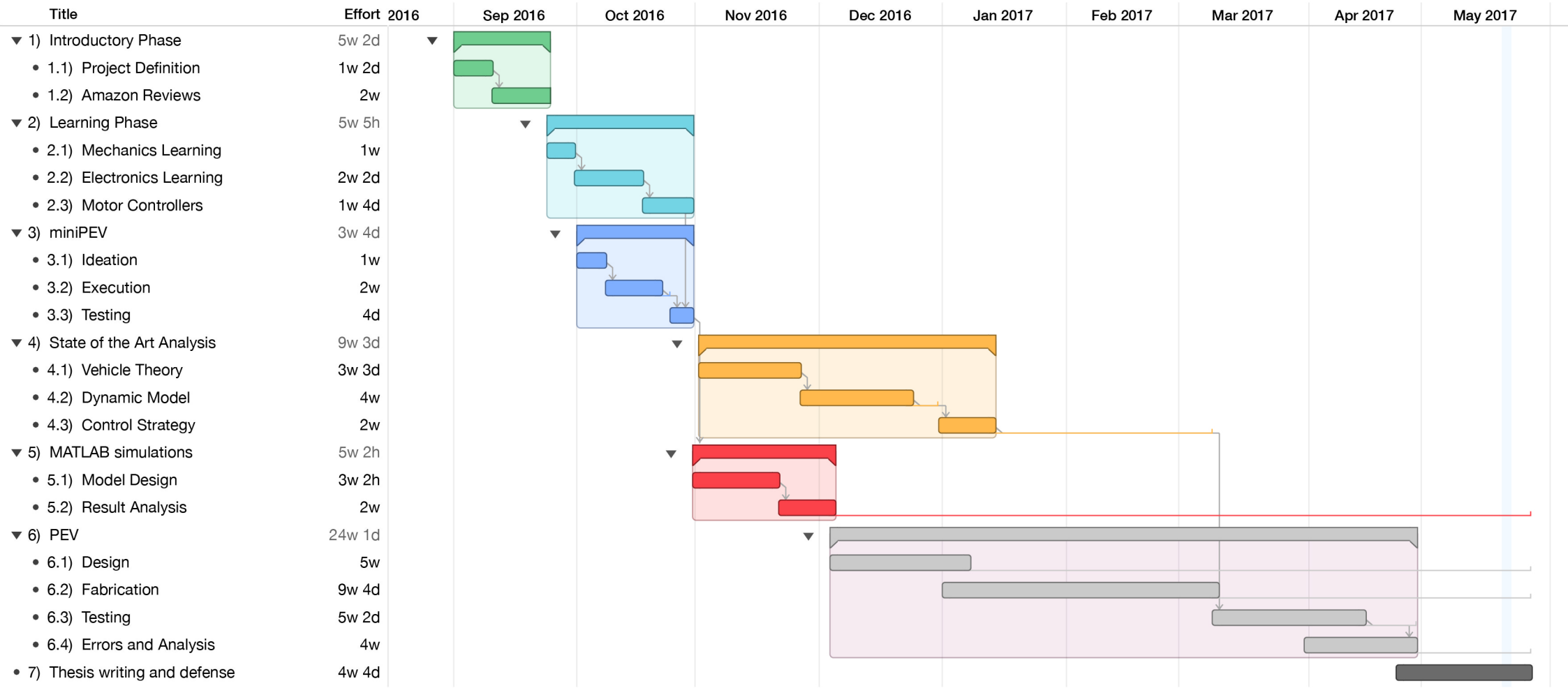
Total Cost

Part	Total price ($)
Materials	565.53
Machining	4980
Labor	17650
Total	**23195.53**

Table 12: Total Cost

Project Timeline

A Gantt chart with the main tasks of the project has been included.


Title
Effort
2016
Sep 2016
Oct 2016
Nov 2016
Dec 2016
Jan 2017
Feb 2017
Mar 2017
Apr 2017
May 2017
1) Introductory Phase 5w 2d
1.1) Project Definition 1w 2d
1.2) Amazon Reviews 2w
2) Learning Phase 5w 5h
2.1) Mechanics Learning 1w
2.2) Electronics Learning 2w 2d
2.3) Motor Controllers 1w 4d
3) miniPEV 3w 4d
3.1) Ideation 1w
3.2) Execution 2w
3.3) Testing 4d
4) State of the Art Analysis 9w 3d
4.1) Vehicle Theory 3w 3d
4.2) Dynamic Model 4w
4.3) Control Strategy 2w
5) MATLAB simulations 5w 2h
5.1) Model Design 3w 2h
5.2) Result Analysis 2w
6) PEV 24w 1d
6.1) Design 5w
6.2) Fabrication 9w 4d
6.3) Testing 5w 2d
6.4) Errors and Analysis 4w
7) Thesis writing and defense 4w 4d

8. Conclusions and Future Work

This final chapter concludes the thesis and summarizes the main conclusions and presents a number of thoughts on future work.

The Persuasive Electric Vehicle was conceived to be an **alternative to the car in urban transportation**. This vehicle, along with other strategies, will progressively take the cars out of the urban areas. The PEV finds itself between the bike and the car. Its weight, size and price makes it very attractive for city councils who are willing to reduce pollution and congestion from their cities. As a vehicle of the future, it combines the three main concepts of the future of transportation: **electrification, autonomy an shared economy**.

It is important to highlight that this thesis vehicle had the purpose of **experimenting with an active tilting system**, which accounts for the principal innovation in lightweight vehicles like PEV. Other active systems, like the rear propulsion, the steering and the haptic feedback in the handle bar were additions to complete the active control of all the motions in the vehicle.

Furthermore, the vehicle was designed with a very clear idea in mind: it vehicle had to be **easy to ride for any user**, independent of their age and experience with bicycles, for example. In fact the design and dimensions of the PEV do were thought to not restrict the handling of the vehicle to any user.

This project has focused on the stability of the PEV and from a general perspective, on the **stability and balance of lightweight three wheeler vehicles**. However, the PEV is addressing more issues that the ones related with stability. The autonomy package is upgrading day by day thanks to the improvement in the computational power and price of embedded systems like NVIDIA Jetson TX2. In this way, this vehicle will be completed with the autonomous package that Changing Places group is developing.

Thanks to the group's **modular component philosophy**, the design on the vehicle allows the embodiment of the sensors of the autonomous package (LIDAR, ultrasound sensors and stereo-camera) in the frame of the vehicle. Not to mention that the existing electronics in this vehicle –composed of the two Arduino Megas and the rest of controllers and sensors– can be integrated into the embedded Jetson TX2 by means of the ROS environment in Linux.

Regarding the **research goals** outlined at the beginning of the thesis, the main objective was to design and fabricate a three wheeler vehicle with an incorporated active tilting system. It can be concluded that the main goal of this thesis was overall achieved, but it has to be pointed out that the fact that tests with driver were not performed does not fully complete it. However, the proposed solutions agree with the existing constraints due to the size, the weight and the cost of a lightweight vehicle like PEV.

The **selection of the motors and the mechanical components** was adequate for the limitations of the vehicle. Specifically, regarding the tilting motor, the attached planetary gears were ideal for increasing the output torque, which is the main limitation with Direct Tilting Control systems. This prevented any problem to tilt the vehicle in the curves and to maintain it in the vertical position when standing.

After reviewing the current developments in vehicle modeling, roll control and tire modeling, a **tilting control system** was implemented in the fabricated vehicle. The justification for the selection of DTC rather than STC lays in the range of speeds of the PEV and also in the complexity of the solutions. In this way, the previous work in the field of tilting vehicles cleared the view and showed the limitations and problems of the applied techniques.

Uncertainties such as neglected dynamics or environmental factors were overcome by an stability and a robustness analysis of the control strategy. In fact, these **uncertainties affected the experimental results** as well as the vehicle modeling and simulation.

The vehicle was set up to **test the tilting system** and the rest of the active systems. Even though the results were satisfactory, it was not possible to compare the experimental data with the simulations, due to the lack of tests. In other terms, the tilting mechanism and the design of the front suspension probed to be suitable for the vehicle.

Regarding the **future improvements in the control strategy**, the modeling of the PEV could be improved with real measurements rather than by trusting a computer generated 3D model. The robustness of the control strategy could also improve if the control gains could be calculated in real time based on the signals coming from the vehicle.

Figure 160: Mens et manus

Bibliography

Chapter 1: Introduction

[1] Department of Economical United Nations and Social Affairs. World urbanization prospects: The 2014 revision, highlights, 2014. ISBN: 978-92-1-151517-6. Available at http://esa.un.org/unpd/wup/Highlights/WUP2014-Highlights.pdf.

[2] Society of Automotive Engineers (SAE). Taxonomy and definitions for terms related to on-road motor vehicle automated driving systems, 2014. Available at http://standards.sae.org/j3016_201401/. Summary available at https://www.sae.org/misc/pdfs/automated_driving.pdf.

[3] Leanna Garfield (Business Insider). 12 major cities that are starting to go car-free, 2017. Available at http://www.businessinsider.com/cities-going-car-free-2017-2.

[4] Changing Places group projects. http://cp.media.mit.edu/.

[5] About the MIT Media Lab. https://www.media.mit.edu/about/mission-history/.

[6] Changing Places group principal investigator Kent Larson. https://www.media.mit.edu/people/kll/overview/.

[7] PEV information. http://cp.media.mit.edu/pev/.

[8] Amazon.com product example. https://www.amazon.com/Schwinn-Meridian-Adult-26-Inch-3-Wheel/dp/B01I92S3F4.

[9] Andrea Esuli. Perl scripts for amazon reviews downloader and parser. https://github.com/aesuli/Amazon-downloader.

[10] Julian John McAuley. Cse 190 data mining and predictive analytics. lecture 13, slide63. In *Proceedings of the 22nd international conference on World Wide Web*. UCSD, 2015. Available at http://cseweb.ucsd.edu/~jmcauley/cse190/slides/week8/lecture13.pdf.

[11] Stuart Rose, Dave Engel, Nick Cramer, and Wendy Cowley. *Automatic Keyword Extraction from Individual Documents*. John Wiley Sons, Ltd, 2010. Available at http://dx.doi.org/10.1002/9780470689646.ch1.

[12] Aashutosh Bhatt, Ankit Patel, Harsh Chheda, and Kiran Gawande. Amazon review classification and sentiment analysis. *International Journal of Computer Science and Information Technologies*, 6(6):5107–5110, 2015. Available at http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.736.4819&rep=rep1&type=pdf.

[13] Abhijit Chakankar, Sanjukta Pal Mathur, and Krishna Venuturimilli. Sentiment analysis of usersâĂŹ reviews and comments. Available at https://pdfs.semanticscholar.org/ab8f/62c23b116ab51224e743f4c45871f8bbe995.pdf.

[14] Subhabrata Mukherjee and Pushpak Bhattacharyya. *Feature Specific Sentiment Analysis for Product Reviews*, pages 475–487. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012. Available at http://dx.doi.org/10.1007/978-3-642-28604-9_39.

[15] Callen Rain. Sentiment analysis in amazon reviews using probabilistic machine learning. *Swarthmore College*, 2013. Available at https://pdfs.semanticscholar.org/f0af/e9ea9d286248336ee9dc4e954aecde3475bb.pdf.

[16] Stephen Nurse, Mark Richardson, and Robbie Napper. Tilting human powered trikes: Principles, designs and new developments. In *Australasian Transport Research Forum (ATRF), 37th, 2015, Sydney, New South Wales, Australia*, 2015. Available at http://atrf.info/papers/2015/index.aspx.

[17] Saeedi, Kazemi, and AWT TAG. Stability of three-wheeled vehicles with and without control system. *International Journal of Automotive Engineering*, 3, 2013. Available at http://www.iust.ac.ir/ijae/article-1-180-en.html.

[18] Andrea Festini, Andrea Tonoli, and Enrico Zenerino. *Urban and extra urban vehicles: re-thinking the vehicle design*. INTECH Open Access Publisher, 2011. Available at https://www.intechopen.com/books/howtoreference/new-trends-and-developments-in-automotive-system-engineering/urban-and-extra-urban-vehicles-re-thinking-the-vehicle-design.

Chapter 2: Background

[19] Commuter Cars. Tango - lowest center of gravity of any car. http://www.commutercars.com/.

[20] Renault. Twizy - compact electric vehicle. https://www.renault.co.uk/vehicles/new-vehicles/twizy.html.

[21] General Motors. Lean machine. http://www.carstyling.ru/en/car/1982_gm_lean_machine/.

[22] Mercedes Benz. Life jet f300. http://www.evo.co.uk/mercedes/18241/concept-cars-the-mercedes-benz-f300-life-jet.

[23] University of Bath and BMW. Clever. http://www.bmwblog.com/2009/10/09/bmw-unveils-clever-concept/.

[24] Vehicle Class, C. R. Van Den Brink, and H. M. Kroonen. *AVEC '04 DVC - The banking technology driving the CARVER*, 2004. Available at http://www.carver-technology.com/PDF/brink_AVEC_2004.pdf.

[25] Piaggio. Mp3 hybrid 300ie. http://www.piaggio.com/mp3/en_EN/models/hybrid/hybrid-300IE/.

[26] Adomeit Group GmbH. Veleon. http://www.veleon.de/eng/veleon.htm#vielfalt.

[27] Trego. http://trego-trolley.com/.

[28] Dimitris-Niavis. Kaylad-e. http://newatlas.com/kaylad-2-electric-tricycle/23049/.

[29] Protanium. Carqon. http://www.carqon.com/.

[30] Sang-Gyun So and Dean Karnoop. Active dual mode tilt control for narrow ground vehicles. *Vehicle System Dynamics*, 27(1):19–36, 1997. Available at http://dx.doi.org/10.1080/00423119708969321.

[31] Robin Hibbard and Dean Karnopp. Twenty first century transportation system solutions-a new type of small, relatively tall and narrow active tilting commuter vehicle. *Vehicle system dynamics*, 25(5):321–347, 1996. Available at http://dx.doi.org/10.1080/00423119608968970.

[32] Sang-Gyun So and Dean Karnopp. Switching strategies for narrow ground vehicles with dual mode automatic tilt control. *International Journal of Vehicle Design*, 18(5):518–532, 1997. Url:http://www.inderscienceonline.com/doi/abs/10.1504/IJVD.1997.062071.

[33] Johan J H Berote. Dynamics and control of a tilting three wheeled vehicle. Available at http://opus.bath.ac.uk/24680/, 2010.

[34] Johan Berote, Auguste Van Poelgeest, Jocelyn Darling, Kevin A Edge, and Andrew Plummer. The dynamics of a three-wheeled narrow-track tilting vehicle. In *FISITA World Automotive Congress 2008*, September 2008. Paper number: F2008-SC-032. Available at http://opus.bath.ac.uk/15090/.

[35] R. Rajamani, J. Gohl, L. Alexander, and P. Starr. Dynamics of narrow tilting vehicles. *Mathematical and Computer Modelling of Dynamical Systems*, 9(2):209–231, 2003. Available at http://www.tandfonline.com/doi/abs/10.1076/mcmd.9.2.209.16521.

[36] D Piyabongkarn, T Keviczky, and R Rajamani. Active direct tilt control for stability enhancement of a narrow commuter vehicle. *International Journal of Automotive Technology*, 5(2):77–88, 2004. Available at https://pdfs.semanticscholar.org/b079/59e2ab35f552f1b06209af731e148e32a0ec.pdf.

[37] S. Kidane, R. Rajamani, L. Alexander, P. J. Starr, and M. Donath. Development and experimental evaluation of a tilt stability control system for narrow commuter vehicles. *IEEE Transactions on Control Systems Technology*, 18(6):1266–1279, Nov 2010. Available at http://ieeexplore.ieee.org/document/5356230/.

[38] S. Kidane, L. Alexander, R. Rajamani, P. Starr, and M. Donath. A fundamental investigation of tilt control systems for narrow commuter vehicles. *Vehicle System Dynamics*, 46(4):295–322, 2008. Available at http://www.tandfonline.com/doi/abs/10.1080/00423110701352987.

Chapter 3: Control Strategy

[39] Lama Mourad. *Active lateral acceleration control of a narrow tilting vehicle*. Theses, Ecole des Mines de Nantes, December 2012. Available at https://tel.archives-ouvertes.fr/tel-00787310/file/Mourad_L_12_2012.pdf.

[40] Robin Hibbard and Dean Karnopp. Twenty first century transportation system solutions-a new type of small, relatively tall and narrow active tilting commuter vehicle. *Vehicle system dynamics*, 25(5):321–347, 1996. Available at http://dx.doi.org/10.1080/00423119608968970.

[41] Sang-Gyun So and Dean Karnoop. Active dual mode tilt control for narrow ground vehicles. *Vehicle System Dynamics*, 27(1):19–36, 1997. Available at http://dx.doi.org/10.1080/00423119708969321.

[42] Sang-Gyun So and Dean Karnopp. Switching strategies for narrow ground vehicles with dual mode automatic tilt control. *International Journal of Vehicle Design*, 18(5):518–532, 1997. Url:http://www.inderscienceonline.com/doi/abs/10.1504/IJVD.1997.062071.

[43] D Piyabongkarn, T Keviczky, and R Rajamani. Active direct tilt control for stability enhancement of a narrow commuter vehicle. *International Journal of Automotive Technology*, 5(2):77–88, 2004. Available at https://pdfs.semanticscholar.org/b079/59e2ab35f552f1b06209af731e148e32a0ec.pdf.

[44] S. Kidane, L. Alexander, R. Rajamani, P. Starr, and M. Donath. A fundamental investigation of tilt control systems for narrow commuter vehicles. *Vehicle System Dynamics*, 46(4):295–322, 2008. Available at http://www.tandfonline.com/doi/abs/10.1080/00423110701352987.

[45] S. Kidane, R. Rajamani, L. Alexander, P. J. Starr, and M. Donath. Development and experimental evaluation of a tilt stability control system for narrow commuter vehicles. *IEEE Transactions on Control Systems Technology*, 18(6):1266–1279, Nov 2010. Available at http://ieeexplore.ieee.org/document/5356230/.

[46] CR van den Brink. Realization of high performance man wide vehicles (mwvs) with an automatic active tilting mechanism. In *Proceedings EAEC European Automotive Congress "Vehicle Systems Technology for the Next Century: Conference II–Vehicle Dynamics and Active Safety", Barcelona*, pages 41–49, 1999. Available at http://carver-technology.com/PDF/brink_EAEC_Barcelona_1999.pdf.

[47] CR van den Brink and HM Kroonen. Dynamic vehicle control for enclosed narrow vehicles. In *Volume I EAEC 6th European Congress "Lightweight and small cars: The answer to Future Needs"*, pages 217–226, 1997. Available at http://www.carver-technology.nl/PDF/brink_EAEC_Cernobbio_1997.pdf.

[48] Johan Berote, Auguste Van Poelgeest, Jocelyn Darling, Kevin A Edge, and Andrew Plummer. The dynamics of a three-wheeled narrow-track tilting vehicle. In *FISITA World Automotive Congress 2008*, September 2008. Paper number: F2008-SC-032. Available at http://opus.bath.ac.uk/15090/.

[49] L. Mourad, F. Claveau, and P. Chevrel. Design of a two dof gain scheduled frequency shaped lq controller for narrow tilting vehicles. In *2012 American Control Conference (ACC)*, pages 6739–6744, June 2012. Available at http://ieeexplore.ieee.org/abstract/document/6315042/.

[50] Kemin Zhou, John C. Doyle, and Keith Glover. *Robust and Optimal Control*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1996. ISBN:0-13-456567-3.

[51] Rensselaer Polytechnic Institute Dr. Kevin Craig, Professor of Mechanical Engineering. *Disturbance Response Notes*. Available at http://studylib.net/doc/18342577/disturbance-response.

[52] University of Stuttgart Prof. Dr. Carsten Scherer. *LQ Optimal Control*, 2013. Available at http://www.mathematik.uni-stuttgart.de/studium/infomat/mst/linearekontrolltheorie/folien/lec5.pdf.

[53] Åbo Akademi University Professor Hannu T. Toivonen, Department of Information Technologies. *The H2-optimal control problem*, 2000. Available at http://users.abo.fi/htoivone/courses/robust/rob3.pdf.

[54] W Murray Wonham. *Linear multivariable control a geometric approach; 2nd ed.* Applications of Mathematics. Springer, New York, 1979. Available at http://cds.cern.ch/record/1614618.

[55] Bernard Friedland. *Control System Design - An Introduction to State-Space Methods*. Dover Publications, 1986. Chapter 9. Linear, Quadratic Optimum Control. Available at http://app.knovel.com/hotlink/toc/id:kpCSDAISS1/control-system-design/control-system-design ISBN: 978-0-486-44278-5.

Chapter 4: Small Scale Prototype

[56] Media Lab Director's new motto. Available at https://slice.mit.edu/2014/07/29/deploy-or-die-media-lab-directors-new-motto/.

[57] miniPEV video. Available at https://vimeo.com/188999255.

Chapter 5: Real Scale Prototype

[58] Alejo Avello Iturriagagoitia. *Teoría de máquinas Segunda Edición*. Escuela Superior de Ingenieros Industriales - Universidad de Navarra, 2014. Available at http://hdl.handle.net/10171/34797.

[59] Solidworks Gear Generator. Available at https://www.thingiverse.com/thing:8077.

[60] Ergotec. *The Guide to Cycling Ergonomics*. Wilhelm Humpert GmbH Co., 2012. Available at http://www.hr.ubc.ca/ergonomics/files/Bike-Ergonomics-reduced-size.pdf.

[61] DC motor in rear wheel hub. Available at http://www.ebikeling.com/shop/electric-bicycle-36v-500w-geared-rear-26-kit-option-1.

[62] Rotary Motion Position-Measuring Transmitter. Available at https://www.mcmaster.com/#16695t57/=17znmw6.

Chapter 6: Vehicle and Control Tests

[63] Peter Windes, Mark Archibald, and Bryan Joseph. Experimental determination of bike tire stiffnesses. Proceedings of the 2013 ASEE North-Central Section Conference, 2013. Available at http://people.cst.cmich.edu/yelam1k/asee/proceedings/2013/papers/97.pdf.

Chapter 7: Project Cost & Overview

[64] K.T. Ulrich and S.D. Eppinger. *Product Design and Development*. McGraw-Hill/Irwin series in marketing. McGraw-Hill/Irwin, 2004. Available at https://books.google.co.uz/books?id=Qju5AAAAIAAJ.

Programming Resources

Python Libraries

- pandas http://pandas.pydata.org/
- numpy http://www.numpy.org/
- nltk http://www.nltk.org/
- matplotlib https://matplotlib.org/
- sklearn http://scikit-learn.org/stable/
- gensim https://radimrehurek.com/gensim/
- wordcloud https://pypi.python.org/pypi/wordcloud
- stop words https://pypi.python.org/pypi/stop-words
- rake https://pypi.python.org/pypi/python-rake
- PIL https://pypi.python.org/pypi/PIL
- random https://docs.python.org/2/library/random.html
- os https://docs.python.org/2/library/os.html
- re https://docs.python.org/2/library/re.html
- string https://docs.python.org/2/library/string.html
- operator https://docs.python.org/2/library/operator.html

Arduino Libraries

- Adafruit Sensor https://github.com/adafruit/Adafruit_Sensor
- Average https://github.com/MajenkoLibraries/Average
- BLDC https://github.com/vedderb/bldc
- DigitalWriteFast https://code.google.com/archive/p/digitalwritefast/downloads
- I2C https://github.com/rambo/I2C
- PID https://github.com/br3ttb/Arduino-PID-Library
- Rotary https://github.com/brianlow/Rotary
- Rotary Encoder with Velocity https://code.google.com/archive/p/arduino-rotary-encoder-with-velocity/
- SoftI2CMaster https://github.com/felias-fogg/SoftI2CMaster
- SoftEasyTransfer https://github.com/madsci1016/Arduino-SoftEasyTransfer
- VescUartControl https://github.com/RollingGecko/VescUartControl
- Time https://github.com/PaulStoffregen/Time

Processing Libraries

- arduino https://github.com/geetarista/processing-arduino
- grafica https://github.com/jagracar/grafica
- httprequests https://github.com/runemadsen/HTTP-Requests-for-Processing
- keystone https://github.com/davidbouchard/keystone
- objloader https://github.com/taseenb/OBJLoader

Tools and Components

Machine Shop Tools

- Ultimaker 2+ 3D printer https://ultimaker.com/en/products/ultimaker-2-plus
- Stratasys Objet Eden260VS 3D printer http://www.stratasys.com/3d-printers/design-series/objet-eden260vs
- Sindoh DP200 3D printer http://3dprinter.sindoh.com/es/ProductDP200
- OMAX 2652 JetMachining Center https://www.omax.com/omax-machine/2652
- Baileigh Manual Roll Bender R-M10 http://www.baileigh.com/manual-roll-bender-r-m10
- GCC LaserPro Spirit GLS http://www.gccworld.com/goods.php?act=view&no=20

miniPEV & PEV Components

- Servo Motor SG90 www.micropik.com/PDF/SG90Servo.pdf
- Omron Rotary Encoder E6B2-C https://www.ia.omron.com/products/family/487/
- PowerSonic PS-1290 http://www.power-sonic.com/images/powersonic/sla_batteries/ps_psg_series/12volt/PS1290.pdf
- Pololu 50:1 Micro Metal Gearmotor https://www.pololu.com/product/998
- Miniature Shock Absorbers MA 35 to MA 900 https://www.hydradynellc.com/images/document/19_Miniature%20Shock%20Absorbers%20MA%2035%20to%20MA%20900.pdf
- HC-05 Bluetooth to Serial Port Module https://www.itead.cc/wiki/Serial_Port_Bluetooth_Module_(Master/Slave)_:_HC-05

- HC-06 Bluetooth to Serial Port Module https://www.itead.cc/wiki/HC06_Serial_Bluetooth_Brick
- HWT-1004-7AB Down Tube Pack for 36V e-Bike https://www.evtaiwan.com.tw/en_US/product/info.html?id=C1D0F29F66F5713B87F01A910298F790&currentRow=19&totalCount=44&iv=N&doPath=1
- NIDEC 48R Motor T-IT-N Characteristics http://www.nidec-motor.com/
- Energizer XP8000 http://www.tomshardware.com/reviews/notebook-battery-external-power-supply,2821-9.html
- Electric Bicycle 36V 500W Gearer Rear 26" KIT http://www.ebikeling.com/shop/electric-bicycle-36v-500w-geared-rear-26-kit-option-1

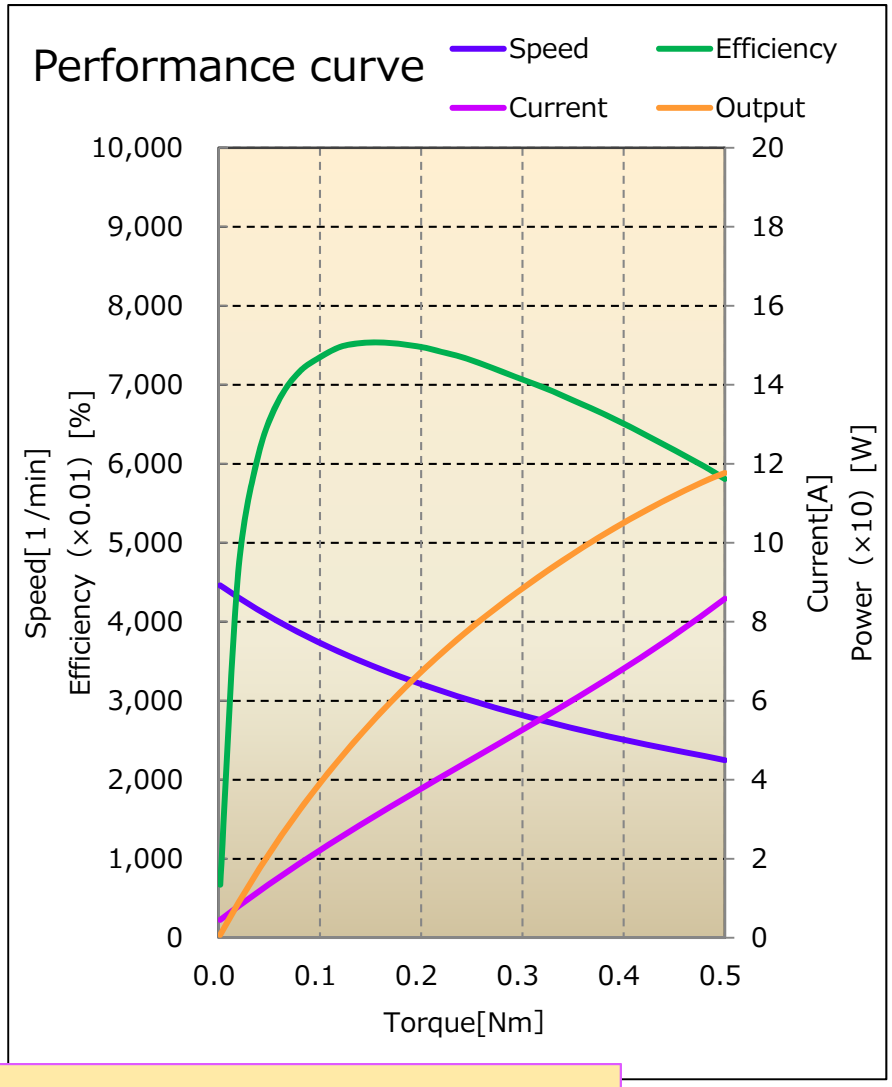
Datasheets

NIDEC 48R Motor

48R Motor T-IT-N Characteristics

Rated output power	**67[W]@DC24[V]**
Rated load torque	0.2[N･m]
Rated speed	3200[1/min]
Rated current	3.8[A]
No load speed	4460[1/min]
No load current	0.5[A]
Instantaneous torque	0.2[Nm]
Encorder	Attached
Short brake	Available
Pulse	24P/R
Bearing	Rolling Bearing
Pole Slot	16P12S
Drive circuit	Attached
Motor weight	30[g]

*Rated load torque is No cooling continuous drive .



○Measurement Condition

・Drive circuit：Square wave drive (Circuit diagram ：G980417200)

Amazon User Review Scripts

Predicting Sentiment and Helpfulness

Topic Modeling

WordCloud Creation

Predicting_Sentiment_and_Helpfulness

May 20, 2017

1 Making predictions over amazon recommendation dataset

The purpose of this analysis is to make up a prediction model where we will be able to predict whether a recommendation is positive or negative. In this analysis, we will not focus on the Score, but only the positive/negative sentiment of the recommendation. In the end, we hope to find a "best" model for predicting the recommendation's sentiment.

1.1 Loading the data

As we only want to get the global sentiment of the recommendations (positive or negative), we will purposefully ignore all Scores equal to 3. If the score id above 3, then the recommendation wil be set to "postive". Otherwise, it will be set to "negative".

The data will be split into an training set and a test set with a test set ratio of 0.2

In [1]:
```
%matplotlib inline

import pandas as pd
import numpy as np
import nltk
import string
import matplotlib.pyplot as plt
import matplotlib as mpl
import numpy as np

from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix
from sklearn import metrics
from sklearn.metrics import roc_curve, auc
from nltk.stem.porter import PorterStemmer
```

In [3]:
```
df = pd.read_table('reviews_2.csv',sep=',',header=None)
df.columns = ['ID','numDate','prod','overall','helpful','votes',
              'date','asin','summary','reviewText']
df.head()
```

Out[3]:

	ID	numDate	prod	overall	helpful	votes	date
0	0	20161030	NaN	2	0	0	on October 30, 2016
1	1	20161030	NaN	4	0	0	on October 30, 2016
2	2	20161029	NaN	5	0	0	on October 29, 2016
3	3	20161028	NaN	5	0	0	on October 28, 2016
4	4	20161027	NaN	5	0	0	on October 27, 2016

```
             asin                                          summary  \
0   A4TDOTZKPA79G  but product arrived in excellent condition.
1   AIR0R7XJECA87                      Not fun to put together
2  A273FNRW9W4RIP                         Get This Cool Trike!
3   AVRR1KJLVE3FF                                     Love it!
4   AN8M8X07W9NXX                                  Sweet ride.

                                          reviewText
0  whobbly to ride.  going around corners VERY ca...
1  Not fun to put together! You have to do a lot ...
2  All Senior Citizens need to own this trike!  V...
3  Needed metric tools to put it together, but on...
4  We bought this, added a boat seat, 80cc gas mo...
```

```
In [4]: messages=df;
        messages["Sentiment"] = messages["overall"].apply(
        lambda score: "positive" if score > 3 else "negative")

        messages["Usefulness"] = (messages["helpful"]/messages["votes"]).apply(
        lambda n: "useful" if n > 0.8 else "useless")
```

2 Extracting features from text data

SciKit cannot work with words, so we'll just assign a new dimention to each word and work with word counts. See more here: http://scikit-learn.org/stable/tutorial/text_analytics/working_with_text_data.html

```
In [8]: from sklearn.feature_extraction.text import
        CountVectorizer, TfidfTransformer

        import re
        import string
        import nltk

        cleanup_re = re.compile('[^a-z]+')
        def cleanup(sentence):
            sentence = sentence.lower()
            sentence = cleanup_re.sub(' ', sentence).strip()
            #sentence = " ".join(nltk.word_tokenize(sentence))
            return sentence

        messages["Summary_Clean"] = messages["summary"].apply(cleanup)
        # messages["Summary_Clean"] = messages["reviewText"].apply(cleanup)

        train, test = train_test_split(messages, test_size=0.2)
        print("%d items in training data, %d in test data
```

```
536 items in training data, 134 in test data
```

```
In [9]: from wordcloud import WordCloud, STOPWORDS
```

```
        # To cleanup stop words, add stop_words = STOPWORDS
        # But it seems to function better without it
        count_vect = CountVectorizer(min_df = 1, ngram_range = (1, 4))
        X_train_counts = count_vect.fit_transform(train["Summary_Clean"])

        tfidf_transformer = TfidfTransformer()
        X_train_tfidf = tfidf_transformer.fit_transform(X_train_counts)

        X_new_counts = count_vect.transform(test["Summary_Clean"])
        X_test_tfidf = tfidf_transformer.transform(X_new_counts)

        y_train = train["Sentiment"]
        y_test = test["Sentiment"]

        prediction = dict()
```

```
In [10]: from wordcloud import WordCloud, STOPWORDS
         stopwords = set(STOPWORDS)

         #mpl.rcParams['figure.figsize']=(8.0,6.0)    #(6.0,4.0)
         mpl.rcParams['font.size']=12                #10
         mpl.rcParams['savefig.dpi']=100             #72
         mpl.rcParams['figure.subplot.bottom']=.1


         def show_wordcloud(data, title = None):
             wordcloud = WordCloud(
                 background_color='white',
                 stopwords=stopwords,
                 max_words=200,
                 max_font_size=40,
                 scale=3,
                 random_state=1
             ).generate(str(data))

             fig = plt.figure(1, figsize=(8, 8))
             plt.axis('off')
             if title:
                 fig.suptitle(title, fontsize=20)
                 fig.subplots_adjust(top=2.3)

             plt.imshow(wordcloud)
             plt.show()

         show_wordcloud(messages["Summary_Clean"])
```


```
In [11]: show_wordcloud(messages[messages.overall == 1]["Summary_Clean"],
title = "Low scoring")
```

Low scoring

```
In [12]: show_wordcloud(messages[messages.overall == 5]["Summary_Clean"],
title = "High scoring")
```

High scoring

```
In [13]: from sklearn.pipeline import FeatureUnion
         from sklearn.base import BaseEstimator, TransformerMixin
         from sklearn.pipeline import Pipeline
         from sklearn.linear_model import LogisticRegression
```

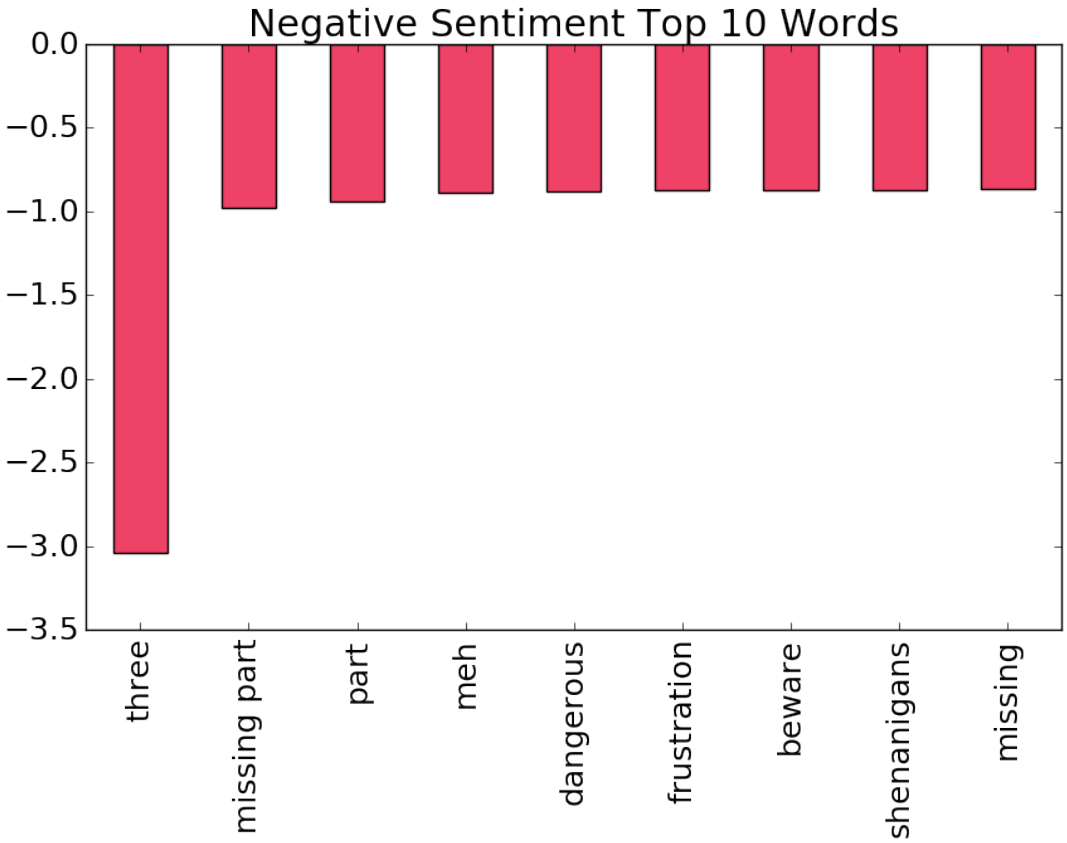
```
        # Useful to select only certain features in a dataset for
        #  forwarding through a pipeline

        class ItemSelector(BaseEstimator, TransformerMixin):
            def __init__(self, key):
                self.key = key
            def fit(self, x, y=None):
                return self
            def transform(self, data_dict):
                return data_dict[self.key]

        train_ufn2, test_ufn2 = train_test_split(messages, test_size=0.2)

        ufn_pipe2 = Pipeline([
           ('union', FeatureUnion(
               transformer_list = [
                   ('summary', Pipeline([
                       ('textsel', ItemSelector(key='Summary_Clean')),
                       ('vect',CountVectorizer(min_df = 1,ngram_range = (1, 4))),
                       ('tfidf', TfidfTransformer())])),
                  ('score', ItemSelector(key=['overall']))
               ],
               transformer_weights = {
                   'summary': 0.2,
                   'score': 0.8
               }
           )),
           ('model', LogisticRegression(C=1e5))
        ])

        ufn_result2 = ufn_pipe2.fit(train_ufn2, train_ufn2["Sentiment"])
```

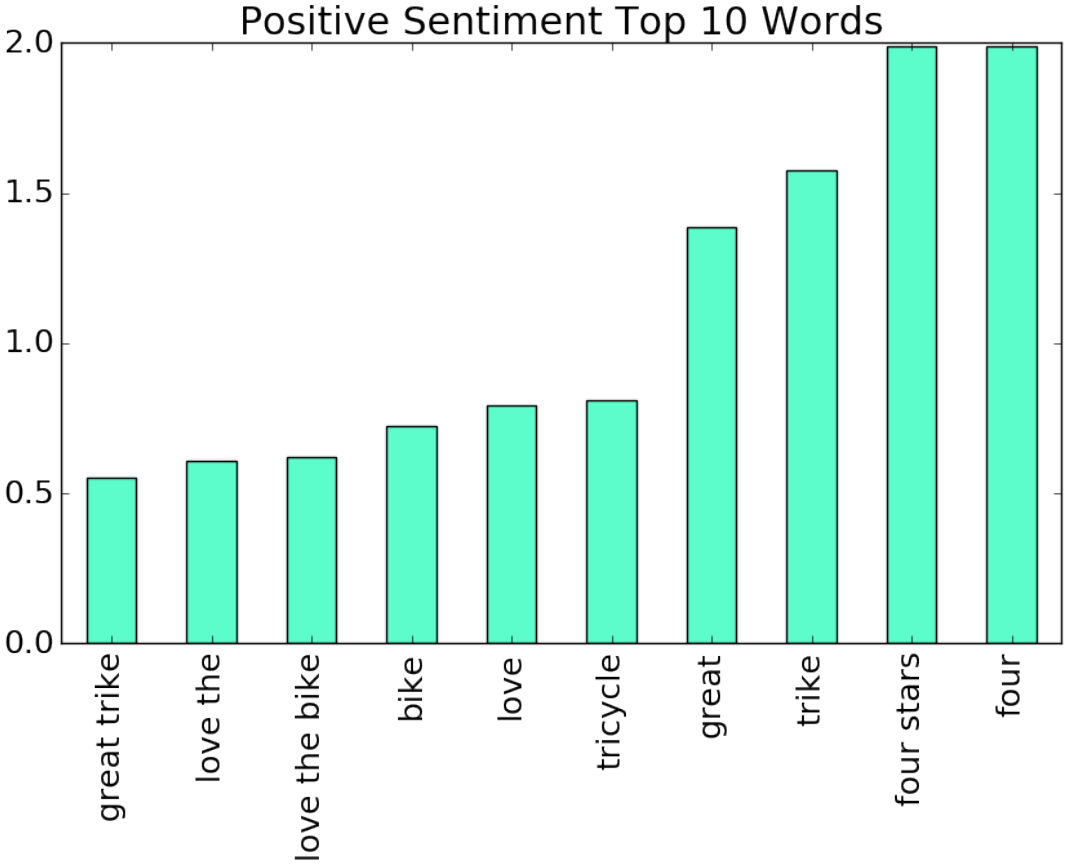
In [ ]:
```
ufn_summary_pipe = next(tr[1] for tr in ufn_result2.named_steps["union"]
ufn_words = ufn_summary_pipe.named_steps['vect'].get_feature_names()
ufn_features = ufn_words
ufn_feature_coefs = pd.DataFrame(
    data = list(zip(ufn_features, ufn_result2.named_steps['model'].
                    coef_[0])),columns = ['feature', 'coef'])
ufn_feature_coefs.sort_values(by='coef')
```

In [21]:
```
df=ufn_feature_coefs.sort_values(by='coef')
type(df)
word_least=df[1:10]
word_most=df[-10:]
```

In [25]:
```
%matplotlib inline
import matplotlib.pyplot as plt
word_least.plot(kind='bar',color='#EE4266',x=word_least['feature'],
legend=False,title='Negative Sentiment Top 10 Words',figsize=(10,6))
word_most.plot(kind='bar',color='#5DFDCB',x=word_most['feature'],
legend=False,title='Positive Sentiment Top 10 Words',figsize=(10,6))
```

Out[25]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x24179707208>
```


Negative Sentiment Top 10 Words
0.0
−0.5
−1.0
−1.5
−2.0
−2.5
−3.0
−3.5
three
missing part
part
meh
dangerous
frustration
beware
shenanigans
missing


Positive Sentiment Top 10 Words
2.0
1.5
1.0
0.5
0.0
great trike
love the
love the bike
bike
love
tricycle
great
trike
four stars
four

Topic Modeling Amazon Reviews

May 20, 2017

```
In [1]: from nltk.tokenize import RegexpTokenizer
        from nltk.corpus import stopwords
        from stop_words import get_stop_words
        from nltk.stem.snowball import SnowballStemmer
        from gensim import corpora, models
        import gensim
```

1 Loading our data

Loading the provided reviews subset CSV into a Pandas dataframe:

```
In [2]: import pandas as pd
        import numpy as np

        #df = pd.read_csv('reviews.csv')

        df = pd.read_table('reviews_2.csv',sep=',',header=None)
        df.columns = ['ID','numDate','prod','overall','helpful','votes',
        'date','asin','summary','reviewText']
```

Now that we have a nice corpus of text, let's go through some of the standard preprocessing required for almost any topic modeling or NLP problem. This approach will involve:

1. Tokenizing: converting a document to its atomic elements
2. Stopping: removing meaningless words
3. Stemming: merging words that are equivalent in meaning

2 Tokenization

We have many ways to segment our document into its atomic elements. To start we'll tokenize the document into words. For this instance we'll use NLTK's `tokenize.regexp` module. You can see how this works in a fun interactive way here: try 'w+' at http://regexr.com/:

```
In [3]: tokenizer = RegexpTokenizer(r'\w+')
```

Running through part of the first review to demonstrate:

```
In [4]: doc_1 = df.reviewText[0]
```

```
In [5]: # Using one of our docs as an example
        tokens = tokenizer.tokenize(doc_1.lower())

        print('{} characters in string vs {} words in a list'.format(len(doc_1),
        print(tokens[:10])
```

```
115 characters in string vs 18 words in a list
['whobbly', 'to', 'ride', 'going', 'around', 'corners', 'very',
'carefully', 'to', 'not']
```

3 Stop Words

Determiners like "the" and conjunctions such as "or" and "for" do not add value to the simple topic model. These types of words are known as stop words and are desired to be removed them from our list of tokens. The definition of a stop work changes depending on the context of the examined documents.

Super list of stop words from the `stop_words` and `nltk` package below.

```
merged_stopwords = [*nltk_stpwd, *stop_words_stpwd]
```

```
In [6]: nltk_stpwd = stopwords.words('english')
        stop_words_stpwd = get_stop_words('en')
        merged_stopwords = list(set(nltk_stpwd + stop_words_stpwd))

        print(len(set(merged_stopwords)))
        print(merged_stopwords[:10])
```

```
207
['i',"hadn't","you'd",'theirs','would','between','t','nor','how','more']
```

```
In [7]:
stopped_tokens=[token for token in tokens if not token in merged_stopwords]
print(stopped_tokens[:10])
```

```
['whobbly', 'ride', 'going', 'around', 'corners', 'carefully', 'tip',
'product', 'arrived', 'excellent']
```

4 Stemming

Stemming allows the reduction of inflectional forms and sometimes derivationally related forms of a word to a common base form. For instance, running and runner to run. Another example:

Amazon's catalog contains bike tires in different sizes and colors ⇒ Amazon catalog contain bike tire in differ size and color

Stemming is a basic and crude heuristic compared to Lemmatization which understands vocabulary and morphological analysis instead of lobbing off the end of words. Essentially Lemmatization removes inflectional endings to return the word to its base or dictionary form of a word, which is defined as the lemma. Great illustrative examples from Wikipedia:

1. *The word "better" has "good" as its lemma. This link is missed by stemming, as it requires a dictionary look-up.*
2. *The word "walk" is the base form for word "walking", and hence this is matched in both stemming and lemmatisation.*
3. *The word "meeting" can be either the base form of a noun or a form of a verb ("to meet") depending on the context, e.g., "in our last meeting" or "We are meeting again tomorrow". Unlike stemming, lemmatisation can in principle select the appropriate lemma depending on the context.*

I start with the common Snowball stemming method, a successor of sorts of the original Porter Stemmer which is implemented in NLTK:

```
In [8]: # Instantiate a Snowball stemmer
        sb_stemmer = SnowballStemmer('english')
```

Note that p_stemmer requires all tokens to be type str. p_stemmer returns the string parameter in stemmed form, so we need to loop through our stopped_tokens:

```
In [9]: stemmed_tokens = [sb_stemmer.stem(token) for token in stopped_tokens]
        print(stemmed_tokens)
```

```
['whobbl', 'ride', 'go', 'around', 'corner', 'care', 'tip', 'product',
'arriv', 'excel', 'condit']
```

5 Putting together a document-term matrix

In order to create an LDA model the 3 steps from above (tokenizing, stopping, stemming) are needed together to create a list of documents (list of lists) to then generate a document-term matrix (unique terms as rows, documents or reviews as columns). This matrix will tell how frequently each term occurs with each individual document.

```
In [10]:
%%time
num_reviews = df.shape[0]
doc_set = [df.reviewText[i] for i in range(num_reviews)]
texts = []

for doc in doc_set:
        # putting our three steps together
        q_tokens = tokenizer.tokenize(word.lower())
        q_stopped_tokens = [token for token in q_tokens
                            if not token in merged_stopwords]
        q_stemmed_tokens=[sb_stemmer.stem(token) for token in q_stopped_tokens]
        # add tokens to list
        texts.append(q_stemmed_tokens[0])
```

```
Wall time: 439 ms
```

```
In [11]: print(texts[0]) # examine review 1
```

```
['whobbl', 'ride', 'go', 'around', 'corner', 'care', 'tip',
'product', 'arriv', 'excel', 'condit']
```

6 Transform tokenized documents into an id-term dictionary

Gensim's Dictionary method encapsulates the mapping between normalized words and their integer ids. Note a term will have an id of some number and in the subsequent bag of words step we can see that id will have a count associated with it.

```
In [12]: # Gensim's Dictionary encapsulates the mapping betweennormalized words
         texts_dict = corpora.Dictionary(texts)
         texts_dict.save('auto_review.dict') # lets save to disk for later use
         # Examine each token's unique id
         print(texts_dict)
```

```
Dictionary(2343 unique tokens: ['faster', 'thick', 'guard', 'tour', 'teas']...)
```

To see the mapping between words and their ids we can use the `token2id` method:

```
In [13]: import operator
         print("IDs 1 through 10: {}".format(sorted(texts_dict.token2id.items(),
         key=operator.itemgetter(1), reverse = False)[:10]))
```

```
IDs 1 through 10: [('tip', 0), ('corner', 1), ('product', 2),
        ('go', 3),('ride', 4), ('condit', 5), ('whobbl', 6),
        ('around', 7), ('arriv', 8), ('care', 9)]
```

```
In [14]: #Guess the original work and examine the count difference between the
        #1 most frequent term and the #10 most frequent term:

         print(df.reviewText.str.contains("balance").value_counts())
         print()
         print(df.reviewText.str.contains("lot").value_counts())
```

```
        False    634
        True      36
        Name: reviewText, dtype: int64

        False    625
        True      45
        Name: reviewText, dtype: int64
```

We have a lot of unique tokens, let's see what happens if we ignore tokens that appear in less than 30 documents or more than 15% documents. Granted this is arbitrary but a quick search shows tons of methods for reducing noise.

```
In [15]: texts_dict.filter_extremes(no_below=30, no_above=0.15) # inlace filter
         print(texts_dict)
         print("top terms:")
         print(sorted(texts_dict.token2id.items(), key=operator.itemgetter(1),
         reverse = False)[:10])
```

```
Dictionary(86 unique tokens: ['two', 'took', 'enjoy', 'wife', 'happi']...)
top terms:
[('easi', 0), ('took', 1), ('enjoy', 2), ('wife', 3), ('happi', 4),
('one', 5),('much', 6), ('nice', 7), ('time', 8), ('thank', 9)]
```

We went from **2343** unique tokens to **86** after filtering. Looking at the top 10 tokens it looks like we got more specific subjects opposed to adjectives.

7 Creating bag of words

Next let's turn `texts_dict` into a bag of words instead. doc2bow converts a `document` (a list of words) into the bag-of-words format (list of `(token_id, token_count)` tuples).

```
In [16]: corpus = [texts_dict.doc2bow(text) for text in texts]
         len(corpus)

Out[16]: 670
```

The corpus is 670 long, the amount of reviews in our dataset and in our dataframe. Let's dump this bag-of-words into a file to avoid parsing the entire text again:

```
In [17]: %%time
         # Matrix Market format
         https://radimrehurek.com/gensim/corpora/mmcorpus.html
         gensim.corpora.MmCorpus.serialize('amzn_auto_review.mm', corpus)

Wall time: 56.5 ms
```

8 Training an LDA model

Training an LDA model using our BOW corpus as training data requires a number of topics, which is set to 10. The number of passes in the training of the model is set to 100 (should be enough).

```
In [18]:
%%time
lda_model = gensim.models.LdaModel(corpus,alpha='auto',
        num_topics=10,id2word=texts_dict, passes=100)

Wall time: 1min 27s
```

9 Inferring Topics

Below are the top 5 words associated with 4 random topics. The float next to each word is the weight showing how much the given word influences this specific topic.

```
In [19]: # For `num_topics` number of topics,
        return `num_words` most significant words
         lda_model.show_topics(num_topics=4,num_words=5)
```

```
Out[19]: [
(0,
'0.138*"year"+0.114*"old"+0.079*"product" + 0.070*"bought" + 0.062*"perfect"'),
(1,
'0.108*"seat"+0.105*"go"+0.075*"got" + 0.062*"basket" + 0.051*"around"'),
(2,
'0.057*"trike"+0.050*"need"+0.048*"fender" + 0.040*"work" + 0.040*"tire"'),
(3,
'0.096*"took"+0.080*"time"+0.065*"new" + 0.057*"littl" + 0.052*"fun"'),
(4,
'0.127*"tricycl"+0.117*"good"+0.104*"qualiti"+0.082*"price"+0.063*"schwinn"'),
(5,
'0.091*"easi"+0.086*"use"+0.058*"realli" + 0.056*"made" + 0.052*"take"'),
(6,
'0.118*"fender"+0.107*"rear"+0.069*"box" + 0.049*"part" + 0.047*"wheel"'),
(7,
'0.100*"nice"+0.048*"want"+0.046*"happi" + 0.043*"one" + 0.043*"sturdi"'),
(8,
'0.152*"wheel"+0.057*"trike"+0.051*"3" + 0.047*"brake" + 0.042*"difficult"'),
(9,
'0.142*"make"+0.128*"thank"+0.080*"easi" + 0.077*"shop" + 0.059*"got"')]
```

LDA is a probabilistic mixture of mixtures (or admixture) model for grouped data. The observed data (words) within the groups (documents) are the result of probabilistically choosing words from a specific topic (multinomial over the vocabulary), where the topic is itself drawn from a document-specific multinomial that has a global Dirichlet prior. This means that words can belong to various topics in various degrees.

10 Querying the LDA Model

Pass an arbitrary string to our model and evaluate what topics are most associated with it.

```
In [20]:
raw_query = 'issue'

query_words = raw_query.split()
query = []
for word in query_words:
        # ad-hoc reuse steps from above
        q_tokens = tokenizer.tokenize(word.lower())
        q_stopped_tokens = [word for word in q_tokens if not word in merged_stop
        q_stemmed_tokens = [sb_stemmer.stem(word) for word in q_stopped_tokens]
        query.append(q_stemmed_tokens[0])
        #different from above, this is not a lists of lists!

print(query)
```

```
['issu']
```

```
In [21]: # translate words in query to ids and frequencies.
         id2word = gensim.corpora.Dictionary()
         _ = id2word.merge_with(texts_dict) # garbage
```

```
In [22]: # translate this document into (word, frequency) pairs
         query = id2word.doc2bow(query)
         print(query)
```

```
[(18, 1)]
```

If we run this constructed query against our trained mode we will get each topic and the likelihood that the `query` relates to that topic. Remember we arbitrarily specified 4 topics when we made the model. When we organize this list to find the most relative topics, we see some intuitive results.

```
In [23]: a = list(sorted(lda_model[query], key=lambda x: x[1],reverse=True))
# sort by the second entry in the tuple
         a
```

```
Out[23]: [(3, 0.66089604585589257),
 (0, 0.042840632894293761),
 (7, 0.041895896144469633),
 (2, 0.039156986576921418),
 (4, 0.038719777833920874),
 (5, 0.038465777946288152),
 (1, 0.038185859271443023),
 (8, 0.035220474260673415),
 (6, 0.03455538873028282),
 (9, 0.030063160485814349)]
```

```
In [24]: lda_model.print_topic(a[0][0]) #most related
```

```
Out[24]: '0.096*"took" + 0.080*"time" + 0.065*"new" + 0.057*"littl" +
         0.052*"fun" + 0.048*"hour" + 0.048*"husband" + 0.042*"two" +
         0.042*"like" + 0.042*"wife"'
```

```
In [25]: lda_model.print_topic(a[-1][0]) #least related
```

```
Out[25]: '0.142*"make" + 0.128*"thank" + 0.080*"easi" + 0.077*"shop" +
         0.059*"got" + 0.045*"take" + 0.043*"back" + 0.040*"pedal" +
         0.033*"handl" + 0.029*"littl"'
```

```
In [26]: import pandas as pd
         word_least = pd.DataFrame(lda_model.show_topic(a[-1][0]),
         columns=['words','count'])
         word_most = pd.DataFrame(lda_model.show_topic(a[0][0]),
         columns=['words','count'])
```

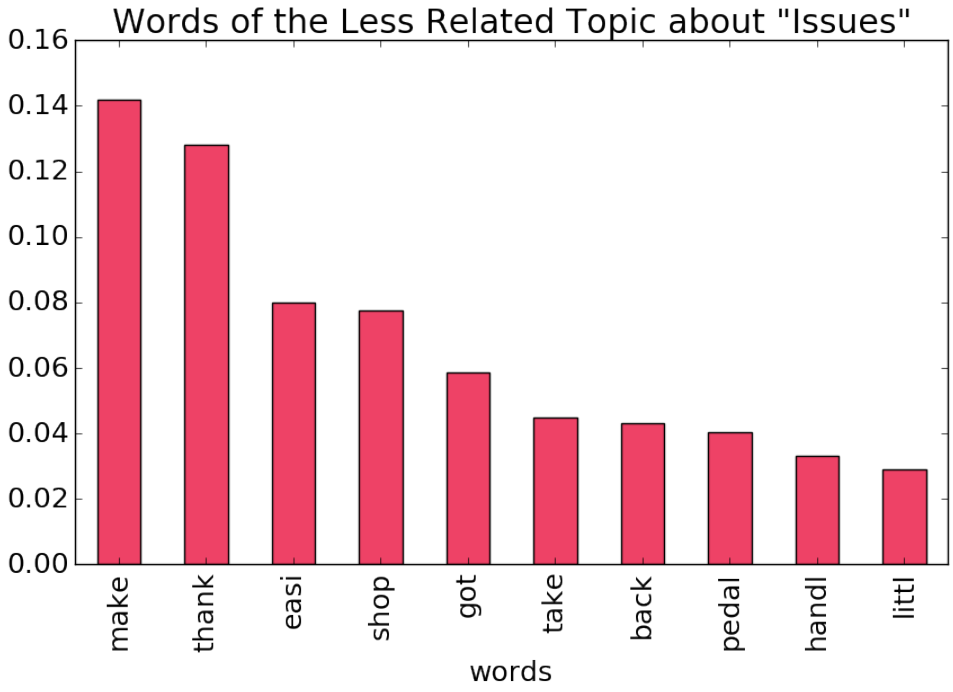
```
In [27]: %matplotlib inline
         import matplotlib.pyplot as plt
         word_least.plot(kind='bar',color='#EE4266',
                 x=word_least['words'], legend=False,
                 title='Words of the Less Related Topic about "Issues"',
                figsize=(10,6))
         word_most.plot(kind='bar',color='#5DFDCB',
                 x=word_most['words'],legend=False,
                 title='Words of the Most Related Topic about "Issues"',
                 figsize=(10,6))
```

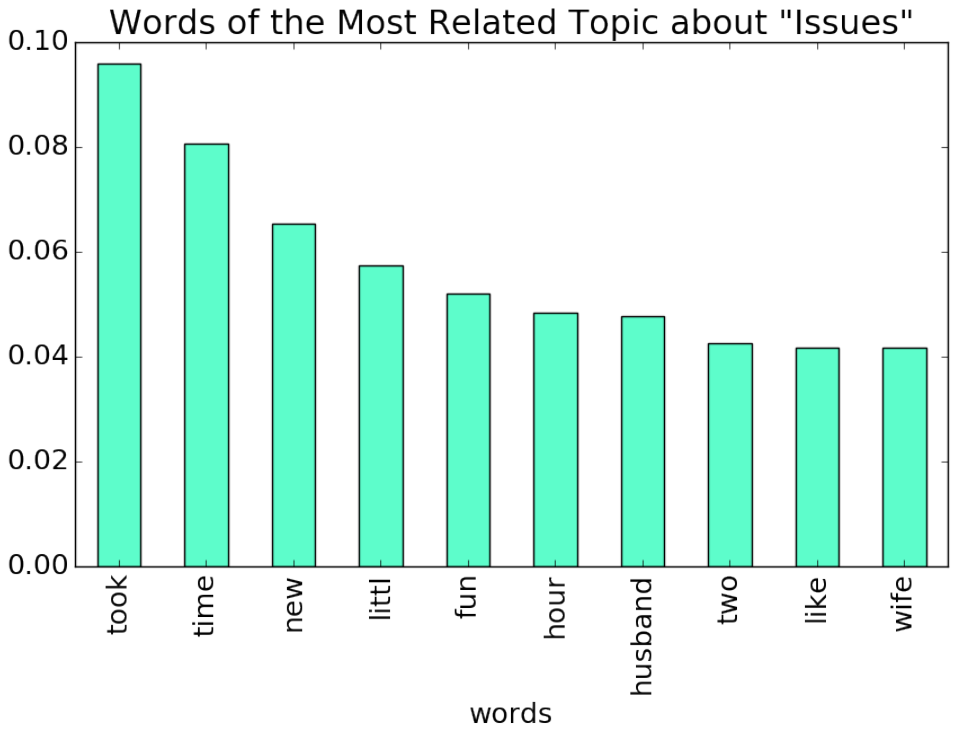
```
Out[27]: <matplotlib.axes._subplots.AxesSubplot at 0x2c6c39e7978>
```

11 Conclusion

Our query "issues" seems to be highly related to the topic number 3, with a correlation of 73%. The rest of topic are mostly unrelated, all below 10% correlation.

By taking these inferred topics and analyzing the sentiment of their corresponding documents (reviews) we can find out what customers are saying (or feeling) about specific products. The LDA model can extract representative statements or quotes, enabling us to summarize customers' opinions about products, perhaps even displaying them on the site. LDA models groups of customers to topics which are groups of products that frequently occur within some customer's orders over time.





Amazon Review - Wordcloud Creation

May 21, 2017

Extracting keywords is one of the most important tasks when working with text. Readers benefit from keywords because they can judge more quickly whether the text is worth reading. Website creators benefit from keywords because they can group similar content by its topics. Algorithm programmers benefit from keywords because they reduce the dimensionality of text to the most important features. And these are just some examples;

By definition, keywords describe the main topics expressed in a document.

1 Import Libraries

```
In [1]: import pandas as pd
        import numpy as np
        from nltk.tokenize import RegexpTokenizer
        from nltk.corpus import stopwords
        from stop_words import get_stop_words
        from nltk.stem.snowball import SnowballStemmer
        from gensim import corpora, models
        import gensim
```

2 Import Reviews Data

```
In [2]: df = pd.read_csv('reviews.csv')
        df.columns = ['ID','numDate','prod','overall','helpful','votes',
                      'date','asin','summary','reviewText']

In [3]: toShow=df[['numDate','overall','helpful','date','summary','reviewText']]
        toShow.head()

Out[3]:    numDate  overall  helpful                 date  \
        0  20161030        2        0  on October 30, 2016
        1  20161030        4        0  on October 30, 2016
        2  20161029        5        0  on October 29, 2016
        3  20161028        5        0  on October 28, 2016
        4  20161027        5        0  on October 27, 2016

                                        summary  \
        0  but product arrived in excellent condition.
        1                       Not fun to put together
        2                          Get This Cool Trike!
        3                                      Love it!
        4                                   Sweet ride.
```

```
                                                   reviewText
0  whobbly to ride.  going around corners VERY ca...
1  Not fun to put together! You have to do a lot ...
2  All Senior Citizens need to own this trike!  V...
3  Needed metric tools to put it together, but on...
4  We bought this, added a boat seat, 80cc gas mo...
```

3 Keyword extraction with Python using RAKE

A typical keyword extraction algorithm has three main components:

1. Candidate selection: extract all possible words, phrases, terms or concepts (depending on the task) that can potentially be keywords.
2. Properties calculation: For each candidate, calculate properties that indicate that it may be a keyword.
3. Scoring and selecting keywords: All candidates can be scored by either combining the properties into a formula, or using a machine learning technique to determine probability of a candidate being a keyword. A score or probability threshold, or a limit on the number of keywords is then used to select the final set of keywords.

Finally, parameters such as the minimum frequency of a candidate, its minimum and maximum length in words, or the stemmer used to normalize the candidates help tweak the algorithm's performance to a specific dataset.

The following algorithm is described in the Text Mining Applications and Theory book by Michael W. Berry:

```
In [4]: import rake
        import operator

        rake_object = rake.Rake("SmartStoplist.txt", 8, 2, 10) #528
        # "SmartStoplist.txt" // "FoxStoplist.txt" // "FrenchStoplist.txt"
        # Now, we have a RAKE object that extracts keywords where:
        # Each word has at least A characters
        # Each phrase has at most B words
        # Each keyword appears in the text at least C times

        text = df["reviewText"].str.cat(sep='\n')
        keywords = rake_object.run(text)
        print("Keywords:", [x[0] for x in keywords])
```

```
Keywords: ['great bike', 'missing parts', 'front fender', 'balance issues',
        'handle bars', 'front tire', 'front brake','fairly easy','nice bike',
        'bike shop', 'wheel bike', 'rear fenders','regular bike','back wheels',
        'highly recommend', 'assembly instructions', 'shipping', 'delivery',
        'excellent', 'assembly', 'comfortable', 'steering', 'christmas',
         'assembled', 'problems', 'recommend', 'tricycle', 'assembling',
         'instructions', 'delivered', 'stability', 'neighborhood', 'daughter',
         'transport', 'scratches', 'difficult', 'exercise','purchase',
         'shopping','directions', 'received', 'replaced','purchased',
         'handlebars','surprised', 'returned', 'assemble','expected',...]
```

4 Wordcloud Creation - v1

```
In [5]: from os import path
        from wordcloud import WordCloud
        # Generate a word cloud image
        wordcloud = WordCloud().generate_from_frequencies(keywords)

        # Display the generated image:
        # the matplotlib way:
        import matplotlib.pyplot as plt
        plt.imshow(wordcloud)
        plt.axis("off")

        # lower max_font_size
        wordcloud = WordCloud(max_font_size=40).
        generate_from_frequencies(keywords)

        plt.figure()
        plt.imshow(wordcloud)
        plt.axis("off")
        plt.show()
```

5 Wordcloud Creation - v2

```
In [6]: import numpy as np
        from PIL import Image
        from os import path
        import matplotlib.pyplot as plt
        import random
```

```
from wordcloud import WordCloud, STOPWORDS


def grey_color_func(word, font_size, position, orientation,
random_state=None,
                    **kwargs):
    return "hsl(0, 0%%, %d%%)" % random.randint(60, 100)

import os
d=os.getcwd()
dd=path.join(d, "mask1.png")

# read the mask image
mask = np.array(Image.open(dd))

# adding movie script specific stopwords
stopwords = set(STOPWORDS)
stopwords=open("SmartStoplist.txt","r")
#stopwords.add("int")
#stopwords.add("ext")

wc = WordCloud(width=1920, height=1080,background_color="black",
max_words=2000, mask=mask, stopwords=stopwords, margin=9,
               random_state=3).
               generate_from_frequencies(keywords) #sortedKeywords

default_colors = wc.to_array()
plt.figure(figsize=(20,10))
plt.imshow(wc.recolor(color_func=grey_color_func,
 random_state=3),interpolation="bilinear")
wc.to_file("wordcloud.png")
plt.axis("off")
plt.show()
```


handle bars
great bike
front brake
balance issues
front fender
front tire
missing parts
fairly easy
nice bike
wheel bike
regular bike
back wheels
bike shop
rear fenders
highly recommend
instructions
handlebars
assembling
comfortable
tricycle
seniors
enjoying
disassemble
exercise
assemble
excellent
difficult
delivered
stability
assembly
directions
special
things
water
ordered
person
recommend
received
enjoy
daughter
steering
shopping
expected
scratches
easily
christmas
perfect
delivery
neighborhood
package
reason
paint
replace
replaced
purchase
ridden
wheels
expect
transport
problems
excellent
delivery
purchased
shipping
surprised
returned
fixed
color
carry
trikes
tools
small
follow
stuff
fenders
bolts
couple
makes
holes
house
ended
garage
tight
hills
trouble
clear
ground
basket
riding

Arduino Code

miniPEV - Arduino UNO

PEV - Arduino MEGA A

PEV - Arduino MEGA B

miniPEV - Arduino UNO

```
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_BNO055.h>
#include <utility/imumaths.h>
#include <Servo.h>
#include <Math.h>

Adafruit_BNO055 bno = Adafruit_BNO055();
Servo myServo1;
Servo myServo2;
int pinS1 = 10;
int pinS2 = 11;

int pinPot = 0;
int pot = 0;

int transistorPin = 3;
int V = 15;

float g = 9.81;
float phi_des = 0;
int delta = 0;
int t1 = 0;
int t2 = 0;

int led = 8;
int threshold = 150;
int deadband = 10;

int buttonPin = 2;         // the number of the input pin

int state = LOW;      // the current state of the output pin
int reading;           // the current reading from the input pin
int previous = HIGH;    // the previous reading from the input pin
long time = 0;         // the last time the output pin was toggled
long debounce = 200;   // the debounce time, increase if the output flickers

int ang_dev = 70;
const float pi = 3.1415;

const int numReadings = 25;
int readings1[numReadings];      // the readings from the analog input
int readIndex1 = 0;              // the index of the current reading
int total1 = 0;                  // the running total
int average1 = 0;                // the average

float readings2[numReadings];      // the readings from the analog input
```

```
int readIndex2 = 0;              // the index of the current reading
float total2 = 0;                  // the running total
float average2 = 0;                // the average

char val;
int pos = LOW;

void setup()
{
  Serial.begin(115200);

  myServo1.attach(pinS1);  myServo1.write(90);
  myServo2.attach(pinS2);  myServo2.write(ang_dev);
  if (!bno.begin())
  {
    Serial.print("Ooops, no BNO055 detected ... Check your wiring or I2C ADDR!");
    while (1);
  }
  //TIMSK0 = 0;
  for (int thisReading1 = 0; thisReading1 < numReadings; thisReading1++) {
    readings1[thisReading1] = 0;
  }
  for (int thisReading2 = 0; thisReading2 < numReadings; thisReading2++) {
    readings2[thisReading2] = 0;
  }

  bno.setExtCrystalUse(true);

  pinMode(buttonPin, INPUT);
  pinMode(led, OUTPUT);
  pinMode(transistorPin, OUTPUT);

  delay(1000);
}

void loop()
{

  reading = digitalRead(buttonPin);

  if (reading == HIGH && previous == LOW && millis() - time > debounce) {
    if (state == HIGH)
      state = LOW;
    else
      state = HIGH;

    time = millis();
  }

```

```
97  digitalWrite(transistorPin, state);
98  digitalWrite(led, state);
99  previous = reading;
100
101 total1 = total1 - readings1[readIndex1];
102 readings1[readIndex1] = analogRead(pinPot);
103 total1 = total1 + readings1[readIndex1];
104 readIndex1 = readIndex1 + 1;
105 if (readIndex1 >= numReadings) {
106   readIndex1 = 0;
107 }
108 average1 = total1 / numReadings;
109
110 pot = average1; //leer valor pot [0-1023]
111 delta = map(pot, 0, 1023, 0, 180); //valor pot [0 - 180]
112 myServo1.write(delta);
113
114 sensors_event_t event;
115 bno.getEventGyroscope(&event);
116
117 total2 = total2 - readings2[readIndex2];
118 readings2[readIndex2] = event.gyro.z;
119 total2 = total2 + readings2[readIndex2];
120 readIndex2 = readIndex2 + 1;
121 if (readIndex2 >= numReadings) {
122   readIndex2 = 0;
123 }
124 average2 = total2 / numReadings;
125
126 phi_des = atan(average2 * V / g) * 180 / pi;
127 //Serial.println(phi_des);
128 myServo2.write(phi_des + ang_dev);
129 delay(50);
130
131 //Read input from bluetooth module:
132 if ( Serial.available())
133 {
134   val = Serial.read();
135
136   //Input key switch
137   switch (val) {
138   case '0':
139     //Code when no key is pressed
140     break;
141   case '1':
142     pos = HIGH;
143     digitalWrite(led, pos);
144     Serial.println("Case 1");
145     //Code when UP key is pressed
```

```
146       break;
147     case '2':
148       pos = LOW;
149       digitalWrite(led, pos);
150       Serial.println("Case 2");
151       //Code when DOWN key is pressed
152       break;
153     case '3':
154       //Code when LEFT key is pressed
155       break;
156     case '4':
157       //Code when RIGHT key is pressed
158       break;
159     default:
160       // default code (should never run)
161       break;
162     }
163   }
164 }
```

PEV - Arduino MEGA A

```
// Arduino Mega for Three Nidec Motors (Tilt, Steer, Handle) & Bluetooth Remote Controller (HC06)

#include "VescUart.h"
#include "datatypes.h"
#include <Wire.h>
#include <Average.h>
#include <SoftwareSerial.h>
#include "I2C_Anything.h"

//Serial: 0-Debug 2-Tilting 1-Steering 3-Handle Bar
int tilt = 2, steer = 1, handle = 3;
const float pi = 3.1415;
#define DEBUG

float delta = 0.0, delta_prev = 0.0, delta2 = 0.0;
float delta_offset = 0, tilt_offset = 0, steer_offset = 0;
int flag1 = 0, flag2 = 0, flag3 = 0, flag4 = 0, flag5 = 0, flag6 = 0;
long tachometer_offset = 0;
struct bldcMeasure handle_bar;
int power_steer = 0;

SoftwareSerial btSerial(10, 11); // RX, TX
char bluetooth = '0';
int remote = 0, btread, demo = 0;

float delta_bt_prev = 0.0, delta_bt = 0.0, theta_bt = 0.0;
volatile boolean haveData = false;
volatile float M_t, delta_imu, theta_imu;

void setup() {
  // SERIAL ---------------------------------
  btSerial.begin(9600);  //BlueTooth
  Serial1.begin(115200); //Tilt
  Serial2.begin(115200); //Steer
  Serial3.begin(115200); //Handle
#ifdef DEBUG
  Serial.begin(115200);
#endif

  // WIRE I2C ---------------------------------
  Wire.begin(8);                // join i2c bus with address #8
  Wire.onReceive(receiveEvent); // register event
  Wire.onRequest(requestEvent); // register event


  // TO DO: INITIAL SETUP----------------------
```

```
  while (haveData)
  {
    Serial.print(M_t); Serial.print('\t');
    Serial.print(delta_imu); Serial.print('\t');
    Serial.print(theta_imu); Serial.print('\t');
    delta_offset = delta_imu;
    tilt_offset = theta_imu;
    haveData = false;
  }

  //  VescUartSetPosition(delta_offset, handle);
  //  delay(2000);
  //  if (VescUartGetValue(handle_bar, handle)) {
  //    tachometer_offset = handle_bar.tachometer;
  //  }

  // ----------------------------------------
}

long prev_time = 0;

void loop() {

  // HANDLE BAR HAPTIC FEEDBACK ----------------
  if (!remote) {
    handle_haptic(0);
    remote = 0;
  }

  // TILTING AND STEERING WITH MASTER DATA
  if (haveData)
  {
    Serial.print(M_t); Serial.print('\t');
    Serial.print(delta_bt); Serial.print('\t');
    Serial.print(delta_imu); Serial.print('\t');
    Serial.print(demo); Serial.print('\t');

    Serial.println(theta_imu); //Serial.print('\t');
    haveData = false;
    // TILTING MOTOR INPUT (I OR ANGLE) -----------
    float I = 0.5 + (M_t / 144) * 33 / 2;
    I = max(I, -5);
    I = min(I, 5);
    if(abs(theta_imu)<0.25){
      VescUartSetCurrent(-I, tilt); //if remote mode, put negative current (-I)
    }

    if(abs(theta_imu)>0.05 && abs(M_t)<5)
    {
```

```
96        VescUartSetPosition(0, tilt);
97      }
98
99      // STEERING MOTOR ANGLE FROM HANDLE BAR INPUT (OR IMU) ----------
100     //Serial.print(delta_imu); Serial.print('\t');
101     if (!remote) {
102       if (fabs(delta_imu) > 0.05 && fabs(delta_imu) < 0.5) {
103         VescUartSetPosition(-delta_imu * 180 / pi, steer);
104       }
105     }
106
107   }
108   // BLUETOOTH COMMUNICATION -------------------
109   //if (remote) {
110     if ( btSerial.available())
111     {
112       remote = 1;
113       btread = 1;
114       bluetooth = btSerial.read();
115
116       switch (bluetooth) {
117         case '0':
118           //Code when no key is pressed
119           //VescUartSetCurrentBrake(0,steer);
120           break;
121         case '1':
122           //Code when UP key is pressed
123           break;
124         case '2':
125           //Code when DOWN key is pressed
126           break;
127         case '3':
128           //Code when LEFT key is pressed
129           if (power_steer) {
130             delta_bt = delta_bt - 10;
131           }
132           break;
133         case '4':
134           //Code when RIGHT key is pressed
135           if (power_steer) {
136             delta_bt = delta_bt + 10;
137           }
138           break;
139         case '5':
140           //Code when SELECT key is pressed
141           if (power_steer == 0) {
142             power_steer = 1;
143             delta_bt = 0;
144           }
```

```
145           else {
146             power_steer = 0;
147             VescUartSetCurrentBrake(3, steer);
148           }
149           break;
150         case '6':
151           //Code when START key is pressed
152           break;
153         case '7':
154           //Code when SQUARE key is pressed
155           theta_bt = theta_bt + 15;
156           theta_bt = min(theta_bt, 60);
157           VescUartSetPosition(theta_bt, tilt);
158           break;
159         case '8':
160           //Code when TRIANGLE key is pressed
161           // VescUartSetPosition(delta_bt, steer);
162           // Include flag for Sponsors Week Demo
163           if(demo==0){
164             demo=1;
165           }
166           else{
167             demo=0;
168           }
169           break;
170         case '9':
171           //Code when X key is pressed
172           theta_bt = theta_bt - 15;
173           theta_bt = max(theta_bt, -60);
174           VescUartSetPosition(theta_bt, tilt);
175           break;
176         case 'A':
177           //Code when CIRCLE key is pressed
178           // BACK TO INITIAL STATE
179           VescUartSetPosition(delta_offset, handle);
180           VescUartSetPosition(tilt_offset, tilt);
181           VescUartSetPosition(steer_offset, steer);
182           delay(1000);
183           // INITIAL OFFSET UPDATE
184           delta_offset = delta_imu;
185           tilt_offset = theta_imu;
186           steer_offset = 0;
187           delta_bt = 0;
188           break;
189         default:
190           break;
191       }
192 
193     //}
```

```
194     delta_bt = max(delta_bt, -40);
195     delta_bt = min(delta_bt, 40);
196     if (delta_bt_prev != delta_bt) {
197       VescUartSetPosition(delta_bt, steer);
198       VescUartSetPosition(-delta_bt, handle);
199     }
200     delta_bt_prev = delta_bt;
201   }
202   //Serial.println();
203 
204   prev_time = millis();
205 }
206 
207 // READ DATA FROM MASTER ARDUINO 2 -----------
208 void receiveEvent(int howMany) {
209   if (howMany >= (sizeof M_t) + (sizeof delta_imu) + (sizeof theta_imu))
210   {
211     I2C_readAnything (M_t);
212     I2C_readAnything (delta_imu);
213     I2C_readAnything (theta_imu);
214     haveData = true;
215   }
216 }
217 
218 // SEND DATA TO MASTER ARDUINO 2 --------------
219 void requestEvent() {
220   if (remote && btread) {
221     Wire.write(bluetooth); // respond with message of 1 bytes as expected by master
222     btread = 0;
223   }
224 }
225 
226 void handle_haptic(int show) {
227 
228   //tachometer_offset = handle_bar.tachometer;
229   if (VescUartGetValue(handle_bar, handle)) {
230 
231     delta = float(handle_bar.tachometer - tachometer_offset) / 17.47; //Change 17.47 for
          ↪ tachometer formulae
232     if (abs(handle_bar.rpm) > 10) {
233       delta2 = delta2 + float(handle_bar.rpm) / 60 * 360 / 1152 * float(millis() -
            ↪ prev_time) / 1000;
234     }
235     // delta3=Euler IMU A
236 
237     if (show) {
238       Serial.print(float(handle_bar.tachometer - tachometer_offset)); Serial.print('\t');
239       Serial.print(handle_bar.dutyCycleNow); Serial.print('\t');
240       Serial.print(handle_bar.avgMotorCurrent); Serial.print('\t');
```

```
241       Serial.print(handle_bar.avgInputCurrent); Serial.print('\t');
242       Serial.print(handle_bar.rpm); Serial.print('\t');
243       Serial.print(handle_bar.inpVoltage); Serial.print('\t');
244       Serial.print(millis()); Serial.print('\t');
245       Serial.print(flag1); Serial.print('\t');
246       Serial.print(flag2); Serial.print('\t');
247       Serial.print(flag3); Serial.print('\t');
248       Serial.print(flag4); Serial.print('\t');
249       Serial.print(flag5); Serial.print('\t');
250       Serial.print(flag6); Serial.print('\t');
251       Serial.print(delta); Serial.print('\t');
252       Serial.println(delta2);
253     }
254     float delta_trans = 5;
255     float delta_limit = 30;
256     float delta_top = 50;
257
258     // Return to center if excessive delta
259     if (delta > delta_top) {
260       flag5 = 1; flag3 = 0; flag1 = 0;
261       VescUartSetPosition(0.0, handle);
262     }
263     else if (flag5 == 1 && abs(float(delta - delta_prev)) == 0) {
264       flag5 = 0;
265       tachometer_offset = handle_bar.tachometer;
266     }
267     // Return to center if excessive delta
268     if (delta < -delta_top) {
269       flag6 = 1; flag4 = 0; flag2 = 0;
270       VescUartSetPosition(0.0, handle);
271     }
272     else if (flag6 == 1 && abs(float(delta - delta_prev)) == 0) {
273       flag6 = 0;
274       tachometer_offset = handle_bar.tachometer;
275     }
276
277     if (flag5 == 0)
278     {
279       //CW Limit Zone
280       if (flag3 == 1 && handle_bar.avgMotorCurrent < 0.25 && handle_bar.rpm > 15)
281       {
282         VescUartSetDuty(0.00, handle);
283       }
284       else if (delta > delta_limit && handle_bar.rpm > 150 && flag3 == 0 )//&& (handle_bar
              ↪ .avgMotorCurrent == 0.0 || flag1 == 1))
285       {
286         VescUartSetDuty(0.00, handle);
287         flag3 = 1;
288       }
```

```
289       else if (flag3 == 1 && (handle_bar.avgMotorCurrent > 0.45 || handle_bar.rpm < 0))
290       {
291         VescUartSetCurrentBrake(0.0, handle);
292         flag3 = 0;
293       }
294       else if (flag3 == 1 && handle_bar.rpm < 50)
295       {
296         VescUartSetCurrentBrake(0.0, handle);
297       }
298     }
299     if (flag6 == 0)
300     {
301       //CCW Limit Zone
302       if (flag4 == 1 && handle_bar.avgMotorCurrent > 0.0 && handle_bar.rpm < 50)
303       {
304         VescUartSetDuty(0.00, handle);
305       }
306       else if (delta < -delta_limit && handle_bar.rpm < -200 && flag4 == 0 && (handle_bar.
              ↪ avgMotorCurrent == 0.0 || flag2 == 1))
307       {
308         VescUartSetDuty(0.00, handle);
309         flag4 = 1;
310       }
311       else if (flag4 == 1 && ((handle_bar.avgMotorCurrent < -0.1 && handle_bar.rpm > 100)
              ↪ || handle_bar.rpm > 300))
312       {
313         VescUartSetCurrentBrake(0.0, handle);
314         flag4 = 0;
315       }
316     }
317     if (flag3 == 0)
318     {
319
320     }
321     if (flag3 == 0 && flag4 == 0 && flag5 == 0 && flag6 == 0)
322     {
323       if (abs(delta) > delta_trans && abs(delta) < delta_limit && handle_bar.rpm > 150 ) {
324         float duty = (delta_limit - delta) / (delta_limit - delta_trans);
325         duty = max(duty, 0); duty = min(duty, 1); int T;
326         if (duty < 0.02) {
327           T = 1000;
328         }
329         else {
330           T = duty * 1000;
331         }
332         VescUartSetDuty(duty, handle);
333         delayMicroseconds(T);
334         VescUartSetDuty(-duty, handle);
335         delayMicroseconds(1);
```

```
336         VescUartSetCurrentBrake(0.0, handle);
337       }
338       else if (abs(delta) > delta_trans && abs(delta) < delta_limit && handle_bar.rpm <
              ↪ -150 ) {
339         float duty = (delta_limit - delta) / (delta_limit - delta_trans);
340         duty = max(duty, 0); duty = min(duty, 1); int T;
341         if (duty < 0.02) {
342           T = 1000;
343         }
344         else {
345           T = duty * 1000;
346         }
347
348         VescUartSetDuty(-duty, handle);
349         delayMicroseconds(T);
350         VescUartSetDuty(duty, handle);
351         delayMicroseconds(1);
352         VescUartSetCurrentBrake(0.0, handle);
353       }
354       //else if (delta > delta_trans && abs(handle_bar.rpm) < 150) {  VescUartSetPosition
              ↪ (0.0,handle);  }
355     }
356     if (flag4 == 0)
357     {
358
359     }
360     //Restart variables only if we read
361     delta_prev = delta;
362     prev_time = millis();
363   }
364   else  {
365     Serial.println("Failed to get data from Handle Bar!");
366   }
367 }
368 void vibrate_A() {
369   prev_time = micros();
370   while ((micros() - prev_time) < 750000)
371   {
372     float duty = 0.5;
373     VescUartSetDuty(duty, handle);
374     delayMicroseconds(10000);
375     VescUartSetDuty(-duty, handle);
376     delayMicroseconds(10000);
377   }
378   VescUartSetCurrentBrake(0.0, handle);
379   //delay(2000);
380 }
381 void vibrate_B() {
382   prev_time = micros();
```

```
383   while ((micros() - prev_time) < 750000)
384   {
385     float duty = 1; //0.4
386     VescUartSetDuty(duty, handle);
387     delayMicroseconds(20000); //7000
388     VescUartSetDuty(-duty, handle);
389     delayMicroseconds(20000);
390   }
391   VescUartSetCurrentBrake(0.0, handle);
392   //delay(2000);
393 }
```

PEV - Arduino MEGA B

```
// Arduino Mega for IMU_A, IMU_B, Throttle, Encoder, Rear Motor & Processing Serial Export (HC05)

#include "VescUart.h"
#include "datatypes.h"
#include <Average.h>
#include <digitalWriteFast.h>
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_BNO055.h>
#include <utility/imumaths.h>
#include <Math.h>
#include "I2C_Anything.h"
#include <SoftwareSerial.h>
SoftwareSerial btSerial(10, 11); // RX, TX
#define DEBUG

#define c_EncoderInterrupt 0
#define c_EncoderPinA 3
#define c_EncoderPinB 2
#define EncoderIsReversed
volatile bool _EncoderBSet;
volatile long _EncoderTicks = 0;
volatile long _EncoderTicks_prev = 0;
float rear_radius = 24.25e-3, v = 0.0;
const float pi = 3.1415;

int processing = 1, calibration_IMU_A = 0, calibration_initial_IMU_A = 0, calibration_IMU_B = 0, calibration_initial_IMU_B = 0;
int print_IMU_A = 0, print_IMU_B = 0, print_control = 1 , print_throttle = 0, print_encoder = 1;
int rear = 1, throttlePin = 0, pot_min = 248, pot_max = 798;
float rear_duty = 0, bt_duty = 0, theta_offset = -0.11 , delta_offset = -2.49, power_rear = 0.0;
char bluetooth;

Average<float> aper(5);
Average<float> thita(5);

// The two BNO055 modules, bnoB has the ADR pin wired to 3.3v to change its i2c address
// Both are wired: SCL to analog 5, SDA to analog 4, VIN to 5v, GRN to ground
Adafruit_BNO055 bnoA = Adafruit_BNO055(-1, BNO055_ADDRESS_A);
Adafruit_BNO055 bnoB = Adafruit_BNO055(-1, BNO055_ADDRESS_B);


typedef struct {
  double q; //process noise covariance
```

```
  double r; //measurement noise covariance
  double x; //value
  double p; //estimation error covariance
  double k; //kalman gain
} kalman_state;

double p = 1, q = 0.02, r = 0.15, x = 0, measurement, k;
kalman_state v_k, a_per_k, phi_p_k, theta_k, theta_p_k, a_per_i_k, delta_k, delta_p_k;


void setup() {
  // SERIAL ---------------------------------
  btSerial.begin(115200);  //BlueTooth
  Serial1.begin(115200); // Rear Motor
#ifdef DEBUG
  Serial.begin(115200);  // Debug
#endif

  // ENCODER --------------------------------
  pinMode(c_EncoderPinA, INPUT);      // sets pin A as input
  digitalWrite(c_EncoderPinA, LOW);  // turn on pullup resistors
  pinMode(c_EncoderPinB, INPUT);      // sets pin B as input
  digitalWrite(c_EncoderPinB, LOW);  // turn on pullup resistors
  attachInterrupt(c_EncoderInterrupt, HandleMotorInterruptA, RISING);

  // THROTTLE -------------------------------
  pinMode(throttlePin, INPUT);
  digitalWrite(throttlePin, LOW);
  pinMode(13, OUTPUT);

  // IMUs CALIBRATION ------------------------
  Wire.begin();

  // IMU A ----------------------------------
  bnoA.begin();
  bnoA.setMode(0X00);
  bnoA.setAccelRange();
  const adafruit_bno055_offsets_t &offsets_type_A = {65526, 65527, 30, 65535, 65535, 0,
      ↪ 65437, 157, 114, 1000, 0};
  //const adafruit_bno055_offsets_t &offsets_type_A =
      ↪ {65526,65527,30,65534,65535,0,65437,157,114,1000,0};

  bnoA.setSensorOffsets(offsets_type_A);
  bnoA.setMode(0X0C);

  if (calibration_IMU_A) {
    while (!bnoA.isFullyCalibrated())
    {
      uint8_t system, gyro, accel, mag = 0;
```

```
91      bnoA.getCalibration(&system, &gyro, &accel, &mag);
92      Serial.print("CALIBRATION␣A:␣Sys=");    Serial.print(system, DEC);
93      Serial.print("␣Gyro=");    Serial.print(gyro, DEC);
94      Serial.print("␣Accel=");   Serial.print(accel, DEC);
95      Serial.print("␣Mag=");     Serial.println(mag, DEC);
96
97      if (calibration_initial_IMU_A) {
98        adafruit_bno055_offsets_t offsets_type_A_2;
99        if (bnoA.getSensorOffsetsAlways(offsets_type_A_2))
100       {
101         Serial.print(offsets_type_A_2.accel_offset_x);    Serial.print(",");
102         Serial.print(offsets_type_A_2.accel_offset_y);    Serial.print(",");
103         Serial.print(offsets_type_A_2.accel_offset_z);    Serial.print(",");
104         Serial.print(offsets_type_A_2.gyro_offset_x);    Serial.print(",");
105         Serial.print(offsets_type_A_2.gyro_offset_y);    Serial.print(",");
106         Serial.print(offsets_type_A_2.gyro_offset_z);    Serial.print(",");
107         Serial.print(offsets_type_A_2.mag_offset_x);    Serial.print(",");
108         Serial.print(offsets_type_A_2.mag_offset_y);    Serial.print(",");
109         Serial.print(offsets_type_A_2.mag_offset_z);    Serial.print(",");
110         Serial.print(offsets_type_A_2.accel_radius);    Serial.print(",");
111         Serial.println(offsets_type_A_2.mag_radius);
112       }
113     }
114   }
115   uint8_t system, gyro, accel, mag = 0;
116   bnoA.getCalibration(&system, &gyro, &accel, &mag);
117   Serial.print("CALIBRATION␣A:␣Sys=");    Serial.print(system, DEC);
118   Serial.print("␣Gyro=");    Serial.print(gyro, DEC);
119   Serial.print("␣Accel=");   Serial.print(accel, DEC);
120   Serial.print("␣Mag=");     Serial.println(mag, DEC);
121 }
122
123 // IMU B ----------------------------------------
124 bnoB.begin();
125 bnoB.setMode(0X00);
126 bnoB.setAccelRange();
127 //const adafruit_bno055_offsets_t &offsets_type_B = {65526, 65527, 30, 65535, 65535, 0,
      ↪ 65437, 157, 114, 1000, 0};//867
128 //const adafruit_bno055_offsets_t &offsets_type_B =
      ↪ {11,4,21,1,1,65535,65437,157,114,1000,0};
129 const adafruit_bno055_offsets_t &offsets_type_B = {65528, 65528, 17, 0, 0, 65535, 65406,
      ↪  150, 114, 1000, 1200};
130 bnoB.setSensorOffsets(offsets_type_B);
131 bnoB.setMode(0X0C);
132
133 if (calibration_IMU_B) {
134   while (!bnoB.isFullyCalibrated())
135   {
136     uint8_t system, gyro, accel, mag = 0;
```

```
137       bnoB.getCalibration(&system, &gyro, &accel, &mag);
138       Serial.print("CALIBRATION␣B:␣Sys=");    Serial.print(system, DEC);
139       Serial.print("␣Gyro=");    Serial.print(gyro, DEC);
140       Serial.print("␣Accel=");   Serial.print(accel, DEC);
141       Serial.print("␣Mag=");     Serial.println(mag, DEC);
142
143       if (calibration_initial_IMU_B) {
144         adafruit_bno055_offsets_t offsets_type_B_2;
145         if (bnoB.getSensorOffsetsAlways(offsets_type_B_2))
146         {
147           Serial.print(offsets_type_B_2.accel_offset_x);    Serial.print(",");
148           Serial.print(offsets_type_B_2.accel_offset_y);    Serial.print(",");
149           Serial.print(offsets_type_B_2.accel_offset_z);    Serial.print(",");
150           Serial.print(offsets_type_B_2.gyro_offset_x);    Serial.print(",");
151           Serial.print(offsets_type_B_2.gyro_offset_y);    Serial.print(",");
152           Serial.print(offsets_type_B_2.gyro_offset_z);    Serial.print(",");
153           Serial.print(offsets_type_B_2.mag_offset_x);    Serial.print(",");
154           Serial.print(offsets_type_B_2.mag_offset_y);    Serial.print(",");
155           Serial.print(offsets_type_B_2.mag_offset_z);    Serial.print(",");
156           Serial.print(offsets_type_B_2.accel_radius);    Serial.print(",");
157           Serial.println(offsets_type_B_2.mag_radius);
158         }
159       }
160     }
161     uint8_t system, gyro, accel, mag = 0;
162     bnoB.getCalibration(&system, &gyro, &accel, &mag);
163     Serial.print("CALIBRATION␣B:␣Sys=");    Serial.print(system, DEC);
164     Serial.print("␣Gyro=");    Serial.print(gyro, DEC);
165     Serial.print("␣Accel=");   Serial.print(accel, DEC);
166     Serial.print("␣Mag=");     Serial.println(mag, DEC);
167   }
168
169   // EXPORT PROCESSING DATA ----------------------------
170   if (processing) {
171     //    btSerial.println("CLEARDATA"); //clears up any data left from previous projects
172     //    btSerial.println("LABEL,..."); //always write LABEL, so excel knows the next
          ↪ things will be the names of the columns (instead of Acolumn you could write
          ↪ Time for instance)
173     //    btSerial.println("RESETTIMER"); //resets timer to 0
174   }
175
176   // Calculate initial angle offsets
177
178   // IMU_A DATA ------------------------------------
179   imu::Quaternion quat_A = bnoA.getQuat();
180   double resA[3];
181   //char RotSeq[12]=[zyx, zyz, zxy, zxz, yxz, yxy, yzx, yzy, xyz, xyx, xzy, xzx];
182   imu::Vector<3> eulerYZX_A = quat_A.quaternion2Euler(7, resA);
183   delay(1000);
```

```
  // IMU_B DATA ----------------------------------- eulerXZY_B.y()
  imu::Quaternion quat_B = bnoB.getQuat();
  double resB[3];
  //char RotSeq[12]=[zyx, zyz, zxy, zxz, yxz, yxy, yzx, yzy, xyz, xyx, xzy, xzx];
  imu::Vector<3> eulerXZY_B = quat_B.quaternion2Euler(1, resB);

  theta_offset = eulerXZY_B.y();
  delta_offset = eulerYZX_A.x() - eulerXZY_B.z();
  Wire.beginTransmission(8); // transmit to device #8
  I2C_writeAnything (0.0);
  I2C_writeAnything (delta_offset - delta_offset);
  I2C_writeAnything (theta_offset - theta_offset);
  Wire.endTransmission();    // stop transmitting
  // ----------------------------------------
  float eulerYZX_A_x_prev = eulerYZX_A.x(), eulerXZY_B_z_prev = eulerXZY_B.z();
  Serial.print(eulerYZX_A.x());  Serial.print('\t');
  Serial.print(eulerXZY_B.z());  Serial.print('\t');
  Serial.print(theta_offset);  Serial.print('\t');
  Serial.print(delta_offset);  Serial.print('\t');

  // KALMAN INITIALIZATION
  v_k = kalman_init(0.005, 0.05, p, x);
  a_per_k = kalman_init(0.01, 0.15, p, x);
  phi_p_k = kalman_init(0.1, r, p, x);
  theta_k = kalman_init(q, r, p, x);
  theta_p_k = kalman_init(q, r, p, x);
  a_per_i_k = kalman_init(q, r, p, x);
  delta_k = kalman_init(q, r, p, x);
  delta_p_k = kalman_init(q, r, p, x);

  digitalWrite(13, HIGH);
  delay(1000);
  digitalWrite(13, LOW);
}
float eulerXZY_B_z, eulerYZX_A_x, eulerYZX_A_x_prev = 0, eulerXZY_B_z_prev = 0, vel = 0.0;
int k_A = 0, k_B = 0;
long delay_loop = 0;
float prev_time = 0, prev_time_enc = micros(), prev_time_a_per = micros();

// GAINS WITHOUT DRIVER AND INVERSE TERM
float k_a_per_c = -0.31, k_phi_p_c = 11.11, k_theta_c = 249.12, k_theta_p_c = 65.51,
    ↪ k_a_per_i_c = -1.00, k_delta_c = 686.17, k_delta_p_c = 123.24;
float k_a_per_v = -0.29, k_phi_p_v = -3.62, k_theta_v = -2.72, k_theta_p_v = -0.49,
    ↪ k_a_per_i_v = 0.00, k_delta_v = -239.04, k_delta_p_v = -54.16;
float k_a_per_v_inv = 0, k_phi_p_v_inv = 0, k_theta_v_inv = 0, k_theta_p_v_inv = 0,
    ↪ k_a_per_i_v_inv = 0.00, k_delta_v_inv = 0, k_delta_p_v_inv = 0;

float a_per, a_per_prev = 0, phi_p , theta , theta_p , a_per_i = 0.0 , delta , delta_p ;

```

```
230 void loop() {
231
232   // ROTARY ENCODER VELOCITY -----------------------
233
234   v = float(_EncoderTicks - _EncoderTicks_prev) / (float(micros() - prev_time_enc) /
        ↪ 1000000) / 1000.0 * 2 * pi * rear_radius; // replace 20 for real delay
235   if (fabs(v) < 0.1) {
236     v = 0.0;
237   }
238   kalman_update(&v_k, v);
239   if (print_encoder) {
240     Serial.print(v_k.x);   Serial.print('\t');
241   }
242   _EncoderTicks_prev = _EncoderTicks;
243   prev_time_enc = micros();
244   //delay(20);
245
246   //  IMU ---------------------------------------
247   //  VECTOR_ACCELEROMETER - m/s^2
248   //  VECTOR_MAGNETOMETER  - uT
249   //  VECTOR_GYROSCOPE     - rad/s
250   //  VECTOR_EULER         - degrees
251   //  VECTOR_LINEARACCEL   - m/s^2
252   //  VECTOR_GRAVITY       - m/s^2
253
254   // IMU_A DATA ------------------------------------
255   imu::Vector<3> linear_accel_A = bnoA.getVector(Adafruit_BNO055::VECTOR_LINEARACCEL);
256   imu::Vector<3> gyro_A = bnoA.getVector(Adafruit_BNO055::VECTOR_GYROSCOPE);
257   imu::Vector<3> euler_A = bnoA.getVector(Adafruit_BNO055::VECTOR_EULER);
258   imu::Vector<3> accel_A = bnoA.getVector(Adafruit_BNO055::VECTOR_ACCELEROMETER);
259   imu::Quaternion quat_A = bnoA.getQuat();
260   double resA[3];
261   //char RotSeq[12]=[zyx, zyz, zxy, zxz, yxz, yxy, yzx, yzy, xyz, xyx, xzy, xzx];
262   imu::Vector<3> eulerYZX_A = quat_A.quaternion2Euler(7, resA);
263
264   // Continuous angle
265   if (eulerYZX_A_x_prev > pi / 2 && eulerYZX_A.x() < -pi / 2) {
266     k_A = k_A + 1;
267   }
268   else if (eulerYZX_A_x_prev < -pi / 2 && eulerYZX_A.x() > pi / 2) {
269     k_A = k_A - 1;
270   }
271   eulerYZX_A_x = eulerYZX_A.x() + k_A * 2 * pi;
272   eulerYZX_A_x_prev = eulerYZX_A.x();
273
274   if (print_IMU_A) {
275     Serial.print(linear_accel_A.x()); Serial.print('\t');
276     Serial.print(linear_accel_A.y()); Serial.print('\t');
277     Serial.print(linear_accel_A.z()); Serial.print('\t');
```

```
278     Serial.print(gyro_A.x()); Serial.print('\t');
279     Serial.print(gyro_A.y()); Serial.print('\t');
280     Serial.print(gyro_A.z()); Serial.print('\t');
281     Serial.print(eulerYZX_A.x()); Serial.print('\t');
282     Serial.print(eulerYZX_A.y()); Serial.print('\t');
283     Serial.print(eulerYZX_A.z()); Serial.print('\t');
284     Serial.print(euler_A.x()); Serial.print('\t');
285     Serial.print(euler_A.y()); Serial.print('\t');
286     Serial.print(euler_A.z()); Serial.print('\t');
287     Serial.print(accel_A.x()); Serial.print('\t');
288     Serial.print(accel_A.y()); Serial.print('\t');
289     Serial.print(accel_A.z()); Serial.print('\t');
290   }
291
292   // IMU_B DATA ------------------------------------
293   imu::Vector<3> linear_accel_B = bnoB.getVector(Adafruit_BNO055::VECTOR_LINEARACCEL);
294   imu::Vector<3> gyro_B = bnoB.getVector(Adafruit_BNO055::VECTOR_GYROSCOPE);
295   imu::Vector<3> euler_B = bnoB.getVector(Adafruit_BNO055::VECTOR_EULER);
296   imu::Vector<3> accel_B = bnoB.getVector(Adafruit_BNO055::VECTOR_ACCELEROMETER);
297   imu::Quaternion quat_B = bnoB.getQuat();
298   double resB[3];
299   //char RotSeq[12]=[zyx, zyz, zxy, zxz, yxz, yxy, yzx, yzy, xyz, xyx, xzy, xzx];
300   imu::Vector<3> eulerXZY_B = quat_B.quaternion2Euler(1, resB);
301
302   // Continuous angle
303   if (eulerXZY_B_z_prev > pi / 2 && eulerXZY_B.z() < -pi / 2) {
304     k_B = k_B + 1;
305   }
306   else if (eulerXZY_B_z_prev < -pi / 2 && eulerXZY_B.z() > pi / 2) {
307     k_B = k_B - 1;
308   }
309   eulerXZY_B_z = eulerXZY_B.z() + k_B * 2 * pi;
310   eulerXZY_B_z_prev = eulerXZY_B.z();
311   if (print_IMU_B) {
312     Serial.print(linear_accel_B.x()); Serial.print('\t');
313     Serial.print(linear_accel_B.y()); Serial.print('\t');
314     Serial.print(linear_accel_B.z()); Serial.print('\t');
315     Serial.print(gyro_B.x()); Serial.print('\t');
316     Serial.print(gyro_B.y()); Serial.print('\t');
317     Serial.print(gyro_B.z()); Serial.print('\t');
318     Serial.print(eulerXZY_B.x()); Serial.print('\t');
319     Serial.print(eulerXZY_B.y()); Serial.print('\t');
320     Serial.print(eulerXZY_B.z()); Serial.print('\t');
321     Serial.print(euler_B.x()); Serial.print('\t');
322     Serial.print(euler_B.y()); Serial.print('\t');
323     Serial.print(euler_B.z()); Serial.print('\t');
324     Serial.print(accel_B.x()); Serial.print('\t');
325     Serial.print(accel_B.y()); Serial.print('\t');
326     Serial.print(accel_B.z()); Serial.print('\t');
```

```
  }
  // THROTTLE VALUE --------------------------------
  rear_duty = float(analogRead(throttlePin) - pot_min) / (pot_max - pot_min);
  if (print_throttle) {
    Serial.print(rear_duty); Serial.print('\t');
    Serial.print(analogRead(throttlePin)); Serial.print('\t');
  }
  // REAR MOTOR POWER ASSIST -----------------------
  if (rear_duty > 0.1 && rear_duty < 0.9) {
    VescUartSetDuty(rear_duty, rear);
  }
  else {
    VescUartSetCurrentBrake(0, rear);
  }
  // CONTROL CALCULATIONS --------------------------------
  float delay_mili = (micros() - prev_time);
  // a_per Kalman Filter ---------------------------------
  if (fabs(linear_accel_B.x()) > 0.5) { //0.5
    kalman_update(&a_per_k, linear_accel_B.x());
  }
  else {
    kalman_update(&a_per_k, 0.0);
  }
  // a_per_i Integration + Kalman Filter-----------------------
  if (abs(a_per_k.x) > 0.02) {
    vel = vel + (micros() - prev_time) / 1000000 * a_per_k.x;
  }
  else {
    vel = 0;
  }
  kalman_update(&a_per_i_k, vel);
  // phi_p Kalman Filter   --------------------------------
  if (fabs(gyro_B.y()) > 0.05) { //0.2
    kalman_update(&phi_p_k, gyro_B.y());
  }
  else {
    kalman_update(&phi_p_k, 0.0);
  }
  // theta Kalman Filter    --------------------------------
  if (fabs(eulerXZY_B.y() - theta_offset) > 0.01) { //0.1
    kalman_update(&theta_k, eulerXZY_B.y() - theta_offset);
  }
  else {
    kalman_update(&theta_k, 0.0);
  }
  // theta_p Kalman Filter  --------------------------------
  if (fabs(gyro_B.z()) > 0.05) { //0.2
    kalman_update(&theta_p_k, gyro_B.z());
  }
```

```
376   else {
377     kalman_update(&theta_p_k, 0.0);
378   }
379   // delta Kalman Filter   ---------------------------------
380   if (fabs(eulerYZX_A_x - eulerXZY_B_z - delta_offset) > 0.05) { //0.2
381     kalman_update(&delta_k, eulerYZX_A_x - eulerXZY_B_z - delta_offset);
382   }
383   else {
384     kalman_update(&delta_k, 0.0);
385   }
386   // delta_p Kalman Filter ---------------------------------
387   if (fabs(gyro_A.x()) > 0.05) { //0.2
388     kalman_update(&delta_p_k, gyro_A.x());
389   }
390   else {
391     kalman_update(&delta_p_k, 0.0);
392   }
393
394   a_per = a_per_k.x; // m/s^2
395   phi_p = phi_p_k.x; // rad/s
396   theta = theta_k.x; // rad
397   theta_p = theta_p_k.x; // rad/s
398   a_per_i = a_per_i_k.x; // m/s
399   delta = delta_k.x; // rad eulerYZX_A.x() - eulerXZY_B.z();
400   delta_p = delta_p_k.x; // rad/s
401
402   a_per_prev = a_per;
403   prev_time_a_per = micros();
404
405   // FEEDBACK CONSTANTS CALCULATION = F(V)
406   float V = v_k.x;
407   float u;
408   if (fabs(V) > 1) {
409     u = -(k_a_per_c + k_a_per_v * V + k_a_per_v_inv / V) * a_per - (k_phi_p_c + k_phi_p_v
          ↪ * V + k_phi_p_v_inv / V) * phi_p - (k_theta_c + k_theta_v * V + k_theta_v_inv /
          ↪  V) * theta - (k_theta_p_c + k_theta_p_v * V + k_theta_p_v_inv / V) * theta_p -
          ↪  (k_a_per_i_c + k_a_per_i_v * V + k_a_per_i_v_inv / V) * a_per_i - (k_delta_c +
          ↪  k_delta_v * V + k_delta_v_inv / V) * delta - (k_delta_p_c + k_delta_p_v * V +
          ↪ k_delta_p_v_inv / V) * delta_p ;
410   }
411   else {
412     u = -k_a_per_c * a_per - k_phi_p_c * phi_p - k_theta_c * theta - k_theta_p_c * theta_p
          ↪  - k_a_per_i_c * a_per_i - k_delta_c * delta - k_delta_p_c * delta_p;
413     u = 0.00;
414   }
415
416   if (print_control) {
417     //    Serial.print(eulerYZX_A.x()); Serial.print('\t');
418     //    Serial.print(eulerXZY_B.z());  Serial.print('\t');
```

```
419
420     Serial.print(a_per); Serial.print('\t');
421     Serial.print(phi_p); Serial.print('\t');
422     Serial.print(theta); Serial.print('\t');
423     Serial.print(theta_p); Serial.print('\t');
424     Serial.print(a_per_i); Serial.print('\t');
425     Serial.print(delta); Serial.print('\t');
426     Serial.print(delta_p); Serial.print('\t');
427     Serial.print(delay_mili); Serial.print('\t');
428     Serial.print(u); Serial.print('\t');
429   }
430
431   // SEND DATA TO SLAVE ARDUINO 1 ------------------
432   Wire.beginTransmission(8); // transmit to device #8
433   I2C_writeAnything (u);
434   I2C_writeAnything (delta);
435   I2C_writeAnything (theta);
436   Wire.endTransmission();    // stop transmitting
437   //delay(5);
438
439   // RECEIVE DATA FROM SLAVE ARDUINO 1 -------------
440   int nbytes = 1;
441   Wire.requestFrom(8, nbytes);    // (request 6 bytes from slave device #8)
442
443   while (Wire.available()) { // slave may send less than requested
444     bluetooth = Wire.read(); // receive a byte as character
445     //Serial.print(bluetooth); Serial.print('\t');
446   }
447   // BLUETOOTH REMOTE CONTROL
448   if (bluetooth == '1')
449   {
450     if (power_rear) {
451       bt_duty = bt_duty + 0.1;
452     }
453   }
454   if (bluetooth == '2')
455   {
456     if (power_rear) {
457       bt_duty = bt_duty - 0.1;
458     }
459   }
460   if (bluetooth == '6')
461   {
462     if (power_rear == 0) {
463       power_rear = 1;
464       bt_duty = 0;
465     }
466     else {
467       power_rear = 0;
```

```
468       VescUartSetCurrentBrake(3, rear);
469     }
470   }
471   if (bluetooth == 'A')
472   {
473     // INITIAL OFFSET UPDATE
474     theta_offset = eulerXZY_B.y();
475     delta_offset = eulerYZX_A.x() - eulerXZY_B.z();
476   }
477   if (power_rear) {
478     VescUartSetDuty(bt_duty, rear);
479   }
480
481   // PROCESSING DATA
482   if (processing) {
483     //btSerial.print("DATA,TIME,TIMER,");
484     if (print_encoder) {
485       btSerial.print(v_k.x); btSerial.print(",");
486     }
487     if (print_IMU_A) {
488       float linear_accel_A_x = linear_accel_A.x(), linear_accel_A_y = linear_accel_A.y(),
          ↪ linear_accel_A_z = linear_accel_A.z();
489       float gyro_A_x = gyro_A.x(), gyro_A_y = gyro_A.y(), gyro_A_z = gyro_A.z();
490       float eulerYZX_A_x = eulerYZX_A.x(), eulerYZX_A_y = eulerYZX_A.y(), eulerYZX_A_z =
          ↪ eulerYZX_A.z();
491       float euler_A_x = euler_A.x(), euler_A_y = euler_A.y(), euler_A_z = euler_A.z();
492       float accel_A_x = accel_A.x(), accel_A_y = accel_A.y(), accel_A_z = accel_A.z();
493
494       btSerial.print(linear_accel_A_x); btSerial.print(",");
495       btSerial.print(linear_accel_A_y); btSerial.print(",");
496       btSerial.print(linear_accel_A_z); btSerial.print(",");
497       btSerial.print(gyro_A_x); btSerial.print(",");
498       btSerial.print(gyro_A_y); btSerial.print(",");
499       btSerial.print(gyro_A_z); btSerial.print(",");
500       btSerial.print(eulerYZX_A_x); btSerial.print(",");
501       btSerial.print(eulerYZX_A_y); btSerial.print(",");
502       btSerial.print(eulerYZX_A_z); btSerial.print(",");
503       btSerial.print(euler_A_x); btSerial.print(",");
504       btSerial.print(euler_A_y); btSerial.print(",");
505       btSerial.print(euler_A_z); btSerial.print(",");
506       btSerial.print(accel_A_x); btSerial.print(",");
507       btSerial.print(accel_A_y); btSerial.print(",");
508       btSerial.print(accel_A_z); btSerial.print(",");
509     }
510     if (print_IMU_B) {
511       float linear_accel_B_x = linear_accel_B.x(), linear_accel_B_y = linear_accel_B.y(),
          ↪ linear_accel_B_z = linear_accel_B.z();
512       float gyro_B_x = gyro_B.x(), gyro_B_y = gyro_B.y(), gyro_B_z = gyro_B.z();
```

```
513       float eulerXZY_B_x = eulerXZY_B.x(), eulerXZY_B_y = eulerXZY_B.y(), eulerXZY_B_z =
              ↪ eulerXZY_B.z();
514       float euler_B_x = euler_B.x(), euler_B_y = euler_B.y(), euler_B_z = euler_B.z();
515       float accel_B_x = accel_B.x(), accel_B_y = accel_B.y(), accel_B_z = accel_B.z();
516
517       btSerial.print(linear_accel_B_x); btSerial.print(",");
518       btSerial.print(linear_accel_B_y); btSerial.print(",");
519       btSerial.print(linear_accel_B_z); btSerial.print(",");
520       btSerial.print(gyro_B_x); btSerial.print(",");
521       btSerial.print(gyro_B_y); btSerial.print(",");
522       btSerial.print(gyro_B_z); btSerial.print(",");
523       btSerial.print(eulerXZY_B_x); btSerial.print(",");
524       btSerial.print(eulerXZY_B_y); btSerial.print(",");
525       btSerial.print(eulerXZY_B_z); btSerial.print(",");
526       btSerial.print(euler_B_x); btSerial.print(",");
527       btSerial.print(euler_B_y); btSerial.print(",");
528       btSerial.print(euler_B_z); btSerial.print(",");
529       btSerial.print(accel_B_x); btSerial.print(",");
530       btSerial.print(accel_B_y); btSerial.print(",");
531       btSerial.print(accel_B_z); btSerial.print(",");
532     }
533     if (print_throttle) {
534       btSerial.print(rear_duty); btSerial.print(",");
535     }
536     if (print_control) {
537       btSerial.print(a_per); btSerial.print(",");
538       btSerial.print(phi_p); btSerial.print(",");
539       btSerial.print(theta); btSerial.print(",");
540       btSerial.print(theta_p); btSerial.print(",");
541       btSerial.print(a_per_i); btSerial.print(",");
542       btSerial.print(delta); btSerial.print(",");
543       btSerial.print(delta_p); btSerial.print(",");
544       btSerial.print(delay_loop); btSerial.print(",");
545       btSerial.print(u); btSerial.print(",");
546     }
547     btSerial.print(micros());
548     btSerial.println();
549   }
550   delay_loop = micros() - prev_time;
551   Serial.print(micros() - prev_time); Serial.print('\t');
552   Serial.println();
553   prev_time = micros();
554   //delay(100);
555 }
556
557 // Interrupt service routines for the left motor's quadrature encoder
558 void HandleMotorInterruptA()
559 {
560   _EncoderBSet = digitalReadFast(c_EncoderPinB);   // read the input pin
```

```
561   // and adjust counter + if A leads B
562 #ifdef EncoderIsReversed
563   _EncoderTicks -= _EncoderBSet ? -1 : +1;
564 #else
565   _EncoderTicks += _EncoderBSet ? -1 : +1;
566 #endif
567 }
568 kalman_state kalman_init(double q, double r, double p, double intial_value)
569 {
570   kalman_state result;
571   result.q = q;
572   result.r = r;
573   result.p = p;
574   result.x = intial_value;
575 
576   return result;
577 }
578 void kalman_update(kalman_state* state, double measurement)
579 {
580   //prediction update
581   //omit x = x
582   state->p = state->p + state->q;
583 
584   //measurement update
585   state->k = state->p / (state->p + state->r);
586   state->x = state->x + state->k * (measurement - state->x);
587   state->p = (1 - state->k) * state->p;
588 }
```

Processing Code

Member Spring Event Visualization

Member Spring Event Visualization

```
import grafica.*;
import processing.serial.*;
import deadpixel.keystone.*;

Keystone ks;
CornerPinSurface surface;

PGraphics offscreen;

Serial myPort;
Table dataTable=new Table();

int numReadings = 100; //keeps track of how many readings you'd like to take before
    ↪ writing the file.
int readingCounter = 0; //counts each reading to compare to numReadings.

String fileName="C:/Users/Inigo/Documents/Processing/Read_Data_wPlot/new.csv";
String val;

GPlot plot, plot1, plot2, plot3, plot4, plot5;
int i = 0; // variable that changes for point calculation
int nPoints = 350; // number of points to display at a time (450)

int colorBG=color(3, 3, 27); // black 0,0,0
int colorText=color(234, 230, 211); //white 255,255,255
int color1=color(252, 142, 255, 80); // pink
int color2=color(111, 208, 255, 80); //blue
int color3=color(181, 174, 190, 80); //grey

int color4=color(255, 131, 0, 80); //orange
int color5=color(0, 255, 131, 80); //green
int color6=color(209, 0, 255, 80); //purple
int gridLinesWidth=1, lineWidth=1, boxLineWidth=2, fontSize=14, pointSize=10;

PShape s;
PFont f1,f2;

void setup()
{

        printArray(Serial.list());
        myPort = new Serial(this, Serial.list()[3], 112500);  //COM15

        dataTable.addColumn("id");
        dataTable.addColumn("year");
        dataTable.addColumn("month");
        dataTable.addColumn("day");
```

```
47          dataTable.addColumn("hour");
48          dataTable.addColumn("minute");
49          dataTable.addColumn("second");
50          dataTable.addColumn("milliseconds");
51
52          dataTable.addColumn("v");
53          dataTable.addColumn("a_per");
54          dataTable.addColumn("phi_p");
55          dataTable.addColumn("theta");
56          dataTable.addColumn("theta_p");
57          dataTable.addColumn("a_per_i");
58          dataTable.addColumn("delta");
59          dataTable.addColumn("delta_p");
60          dataTable.addColumn("delay_loop");
61          dataTable.addColumn("u");
62          dataTable.addColumn("timer");
63
64          //Window settings
65          //size(1280, 720);
66          fullScreen(P3D, 2);
67          //size(800, 600, P3D);
68          background(colorBG);
69          //smooth(2);
70          ks = new Keystone(this);
71          surface = ks.createCornerPinSurface(1920, 1080, 20);
72          offscreen = createGraphics(1920, 1080, P3D);
73
74          // Separation Line
75          offscreen.stroke(color(255, 255, 255, 255));
76          offscreen.line(width/3, 0, width/3, height);
77          //Initial Points
78          GPointsArray points = new GPointsArray(nPoints);
79
80          // calculate initial display points
81          for (i = 0; i < nPoints; i++) {
82            points.add(i, 0);
83          }
84
85          // PLOT ----------------------------------------------------------------
86          plot = new GPlot(offscreen);
87          plot.setPos(width/3, 0); // set the position of to left corner of plot
88          plot.setOuterDim(width/3, height/3);
89          plot.setMar(30, 50, 40, 30);// bottom, left, top, right 60,70,40,30
90          plot.setYLim(0, 10); // set y limits
91          plot.setTitleText("SPEED (m/s)");
92          plot.setPoints(points);
93
94          plot.setBgColor(colorBG);
95          plot.setBoxBgColor(colorBG);
```

```
96          plot.setBoxLineColor(color(48, 48, 48));
97          plot.setBoxLineWidth(boxLineWidth);
98
99          plot.setGridLineColor(color3);
100         plot.setGridLineWidth(gridLinesWidth);
101
102         plot.setLineColor(color3);
103         plot.setLineWidth(lineWidth);
104
105         plot.setPointSize(pointSize);
106         plot.setPointColor(color1);
107         plot.setAllFontProperties("Montserrat", color(255, 255, 255, 200), fontSize);//"
                ↪ SansSerif"
108
109         s = loadShape("Untitled_7.obj");
110         f1 = createFont("Montserrat",50);
111         f2 = createFont("Montserrat",30);
112         textMode(SHAPE);
113
114         myPort.bufferUntil(10);
115 }
116
117 void serialEvent(Serial myPort){
118 }
119
120 void draw()
121 {
122         if (myPort.available()>0){
123         val = myPort.readStringUntil('\n');
124         if (val!= null) { //We have a reading! Record it.
125                 val = trim(val); //gets rid of any whitespace or Unicode nonbreakable
                        ↪ space
126                 println(val); //Optional, useful for debugging. If you see this, you know
                        ↪ data is being sent. Delete if  you like.
127                 float sensorVals[] = float(split(val, ',')); //parses the packet from
                        ↪ Arduino and places the valeus into the sensorVals array. I am
                        ↪ assuming floats. Change the data type to match the datatype coming
                        ↪ from Arduino.
128                 println(sensorVals);
129                 if (sensorVals.length==11) {
130                 TableRow newRow = dataTable.addRow(); //add a row for this new reading
131                 newRow.setInt("id", dataTable.lastRowIndex());//record a unique identifier
                        ↪  (the row's index)
132
133         //record time stamp
134         newRow.setInt("year", year());
135         newRow.setInt("month", month());
136         newRow.setInt("day", day());
137         newRow.setInt("hour", hour());
```

```
138         newRow.setInt("minute", minute());
139         newRow.setInt("second", second());
140         newRow.setInt("milliseconds", millis());
141 
142         newRow.setFloat("v", sensorVals[0]);
143         newRow.setFloat("a_per", sensorVals[1]);
144         newRow.setFloat("phi_p", sensorVals[2]);
145         newRow.setFloat("theta", sensorVals[3]);
146         newRow.setFloat("theta_p", sensorVals[4]);
147         newRow.setFloat("a_per_i", sensorVals[5]);
148         newRow.setFloat("delta", sensorVals[6]);
149         newRow.setFloat("delta_p", sensorVals[7]);
150         newRow.setFloat("delay_loop", sensorVals[8]);
151         newRow.setFloat("u", sensorVals[9]);
152         newRow.setFloat("timer", sensorVals[10]);
153 
154         readingCounter++;
155 
156         if (readingCounter % numReadings ==0)
157         {
158           saveTable(dataTable, fileName);
159         }
160 
161         offscreen.beginDraw();
162         offscreen.background(colorBG);
163         offscreen.textFont(f1);
164         offscreen.fill(color(255, 255, 255, 200));
165         int sep=60, h_ini=80, w_ini=30;
166         offscreen.text("PERSUASIVE_ELECTRIC", w_ini, h_ini);
167         offscreen.text("VEHICLE", w_ini, h_ini+sep);
168         offscreen.textFont(f2);
169         offscreen.fill(color(255, 255, 255, 200));
170         offscreen.text("+_ACTIVE_TILTING_SYSTEM", w_ini, h_ini+2*sep);
171         offscreen.text("+_POWER_ASSIST", w_ini, h_ini+3*sep);
172         offscreen.text("+_STEER_BY_WIRE", w_ini, h_ini+4*sep);
173         offscreen.text("+_HAPTIC_FEEDBACK_IN_HANDLE_BAR", w_ini, h_ini+5*sep);
174 
175         offscreen.ambientLight(128, 128, 180);
176         offscreen.directionalLight(100, 100, 255, 0, 1, 0);
177         offscreen.pointLight(255, 255, 255,width/8, 3*height/4,-100);
178 
179         offscreen.translate(width/8, 3*height/4,-100);
180         offscreen.rotateZ(radians(180));
181         offscreen.rotateX(radians(10));
182         offscreen.rotateY(radians(-20));
183         offscreen.rotateY(radians(35));
184         offscreen.scale(0.35,0.35,0.35);
185         offscreen.shape(s, 0, 0);
186 
```

```
187         offscreen.endDraw();
188 
189         i++;
190         //Plot Data ------------------------------------------------------------
191         plot.addPoint(i, sensorVals[0]);
192         plot.setXLim(i-nPoints, i);
193         plot.beginDraw();
194         plot.drawBackground();
195         plot.drawBox();
196         plot.drawGridLines(0);
197         plot.drawYAxis();
198         plot.drawTitle();
199         plot.drawPoints();
200         plot.endDraw();
201         plot.removePoint(0);
202 
203         background(0);
204         surface.render(offscreen);
205       }
206     }
207   }
208 }
209 void keyPressed() {
210   switch(key) {
211   case 'c':
212     ks.toggleCalibration();
213     break;
214 
215   case 'l':
216     ks.load();
217     break;
218 
219   case 's':
220     ks.save();
221     break;
222   }
223 }
```

Matlab Code

Front Suspension Simulations

Stability and Uncertainty

Front Suspension Simulations

Initial Data

```

global q qini n m
m=39; n=41;

kingpin=70*pi/180; %
track=850e-3; %
h_ground=150e-3; %

    L_6_7=250e-3; %
    L_6_8=2*L_6_7;
    L_14_15=L_6_7;
    L_14_16=L_6_8;

    L_4_5=60e-3; %
    L_3_5=50e-3; %
    L_3_4=L_3_5 + L_4_5;
    L_12_13=L_4_5;
    L_11_13=L_3_5;
    L_11_12=L_3_4;


    a=L_6_7-L_3_5*sin(kingpin)-h_ground;
    L_2_10=31e-3; %
    L_1_9=51e-3; %

    L_1_2=sqrt((L_3_4*sin(kingpin))^2+((L_1_9-L_2_10)/2)^2);
    L_A_1=sqrt(a^2+(L_1_9/2)^2);
    L_A_2=sqrt((L_3_4*sin(kingpin)+a)^2+(L_2_10/2)^2);
    L_A_9=L_A_1;
    L_A_10=L_A_2;
    L_9_10=L_1_2;

    L_5_7=75e-3; %
    L_13_15=L_5_7;

    L_1_3=(track/2)-L_5_7-(L_1_9/2)+L_3_5*cos(kingpin); %
    L_2_4=(track/2)-L_5_7-(L_2_10/2)-L_4_5*cos(kingpin); %
    L_9_11=L_1_3;
    L_10_12=L_2_4;

    c=100e-3; %
    L_A_17=50e-3; %
    L_1_18=c;
    L_9_19=L_1_18;

```

```
46
47      x_A=0; y_A=0;
48      x_1=x_A-L_1_9/2; y_1=y_A+a;
49      x_2=x_1+((L_1_9-L_2_10)/2); y_2=y_1+L_3_4*sin(kingpin);
50      x_3=x_1-L_1_3; y_3=y_1;
51      x_4=x_2-L_2_4; y_4=y_2;
52      x_5=x_3+L_3_5*cos(kingpin); y_5=y_3+L_3_5*sin(kingpin);
53      x_6=x_5-L_5_7; y_6=y_5-L_6_7;
54      x_7=x_6; y_7=y_5;
55      x_8=x_7; y_8=y_7+L_6_7;
56
57      x_9=-x_1; y_9=y_1;
58      x_10=-x_2; y_10=y_2;
59      x_11=-x_3; y_11=y_3;
60      x_12=-x_4; y_12=y_4;
61      x_13=-x_5; y_13=y_5;
62      x_14=-x_6; y_14=y_6;
63      x_15=-x_7; y_15=y_7;
64      x_16=-x_8; y_16=y_8;
65
66      x_17=x_A; y_17=y_A-L_A_17;
67      x_18=x_1-L_1_18; y_18=y_1;
68      x_19=x_9+L_9_19; y_19=y_9;
69
70      s_1=sqrt((x_17-x_18)^2 + (y_17-y_18)^2); s_10=s_1;
71      s_2=sqrt((x_17-x_19)^2 + (y_17-y_19)^2); s_20=s_2;
72
73      phi=atan((2*a)/L_1_9);
74      phi=acos(((x_9-x_A)*(x_17-x_A)+(y_9-y_A)*(y_17-y_A))/(L_A_9*L_A_17));
75
76      x_B=x_6-400e-3;
77      y_B=y_6;
78      x_C=x_14+400e-3;
79      y_C=y_14;
80
81  qini=[x_1 y_1 x_2 y_2 x_3 y_3 x_4 y_4 x_5 y_5 x_6 y_6 x_7 y_7 x_8 y_8 x_9 y_9 x_10 y_10
        ↪ x_11 y_11 x_12 y_12 x_13 y_13 x_14 y_14 x_15 y_15 x_16 y_16 x_17 y_17 x_18 y_18
        ↪ x_19 y_19 s_1 s_2 phi]';
82
83  barras={'A',1,'A',9,1,2,2,10,9,10,1,3,2,4,3,4,5,7,6,8,9,11,10,12,11,12,13,15,14,16};
84  g=9.81;
85  b_1=15e-3;
86  b_2=50e-3;
87  density=2700;
88  masa=density*b_1*b_2*[L_A_1,L_A_9,L_1_2,L_2_10,L_9_10,L_1_3,L_2_4,L_3_4,L_5_7,L_6_8,L_9_11
        ↪ ,L_10_12,L_11_12,L_13_15,L_14_16];
89  pos=[1/2,0;  1/2,0;   1/2,0;   1/2,0;   1/2,0;   1/2,0;   1/2,0;   1/2,0;   1/2,0;
        ↪ 1/2,0;   1/2,0;   1/2,0;   1/2,0;   1/2,0;   1/2,0;   ];
90
```

```
F_apl=[0 -masa(1)*g;    0 -masa(2)*g;    0 -masa(3)*g;    0 -masa(4)*g;    0 -masa(5)*g;    0 -masa(6)*g;    0 -masa(7)*g;    0 -masa(8)*g;    0 -masa(9)*g;    0 -masa(10)*g;    0 -masa(11)*g;    0 -masa(12)*g;    0 -masa(13)*g;    0 -masa(14)*g;    0 -masa(15)*g;]';
save('Constants.mat','m','n','barras','masa','pos','F_apl','s_10','s_20','L_A_1','L_A_9','L_A_2','L_A_10','L_1_2','L_9_10','L_2_10','L_A_17','L_1_18','L_1_3','L_9_19','L_9_11','L_2_4','L_10_12','L_3_5','L_3_4','L_11_13','L_11_12','L_5_7','L_6_7','L_4_5','L_13_15','L_14_15','L_12_13','L_6_8','L_14_16','x_A','y_A','x_B','y_B','x_C','y_C','kingpin','track');

q=qini;
phi_val=phi;
syms x_1 y_1 x_2 y_2 x_3 y_3 x_4 y_4 x_5 y_5 x_6 y_6 x_7 y_7 x_8 y_8 x_9 y_9 x_10 y_10 x_11 y_11 x_12 y_12 x_13 y_13 x_14 y_14 x_15 y_15 x_16 y_16 x_17 y_17 x_18 y_18 x_19 y_19 s_1 s_2 phi
vars=[x_1 y_1 x_2 y_2 x_3 y_3 x_4 y_4 x_5 y_5 x_6 y_6 x_7 y_7 x_8 y_8 x_9 y_9 x_10 y_10 x_11 y_11 x_12 y_12 x_13 y_13 x_14 y_14 x_15 y_15 x_16 y_16 x_17 y_17 x_18 y_18 x_19 y_19 s_1 s_2 phi];

r=ConstraintEq; r=matlabFunction(r);
r2=ConstraintEq;
J=jacobian(r,vars); J=matlabFunction(J);

syms x_1(t) y_1(t) x_2(t) y_2(t) x_3(t) y_3(t) x_4(t) y_4(t) x_5(t) y_5(t) x_6(t) y_6(t) x_7(t) y_7(t) x_8(t) y_8(t) x_9(t) y_9(t) x_10(t) y_10(t) x_11(t) y_11(t) x_12(t) y_12(t) x_13(t) y_13(t) x_14(t) y_14(t) x_15(t) y_15(t) x_16(t) y_16(t) x_17(t) y_17(t) x_18(t) y_18(t) x_19(t) y_19(t) s_1(t) s_2(t) phi(t)
varst=[x_1(t) y_1(t) x_2(t) y_2(t) x_3(t) y_3(t) x_4(t) y_4(t) x_5(t) y_5(t) x_6(t) y_6(t)  x_7(t) y_7(t) x_8(t) y_8(t) x_9(t) y_9(t) x_10(t) y_10(t) x_11(t) y_11(t) x_12(t) y_12(t) x_13(t) y_13(t) x_14(t) y_14(t) x_15(t) y_15(t) x_16(t) y_16(t) x_17(t) y_17(t) x_18(t) y_18(t) x_19(t) y_19(t) s_1(t) s_2(t) phi(t)]';
aux=sym2cell(sort(varst));
J2=J(aux{1:n});
J2=diff(J2,t);

save('Initial_data.mat','x_1','y_1','x_2','y_2','x_3','y_3','x_4','y_4','x_5','y_5','x_6','y_6','x_7','y_7','x_8','y_8','x_9','y_9','x_10','y_10','x_11','y_11','x_12','y_12','x_13','y_13','x_14','y_14','x_15','y_15','x_16','y_16','x_17','y_17','x_18','y_18','x_19','y_19','s_1','s_2','phi','vars','varst','r','r2','J','J2');%,'x_1(t)','y_1(t)','x_2(t)','y_2(t)','x_3(t)','s(t)','phi(t)','varst')
```

Constraint Equations

```
function r=ConstraintEq

    load('Constants.mat');

    syms x_1 y_1 x_2 y_2 x_3 y_3 x_4 y_4 x_5 y_5 x_6 y_6 x_7 y_7 x_8 y_8 x_9 y_9 x_10 y_10
        ↪  x_11 y_11 x_12 y_12 x_13 y_13 x_14 y_14 x_15 y_15 x_16 y_16 x_17 y_17 x_18
        ↪ y_18 x_19 y_19 s_1 s_2 phi
    %r=zeros(m,1);

    r(1)=(x_1-x_A)^2+(y_1-y_A)^2-L_A_1^2;
    r(2)=(x_9-x_A)^2+(y_9-y_A)^2-L_A_9^2;
    r(3)=(x_2-x_A)^2+(y_2-y_A)^2-L_A_2^2;
    r(4)=(x_10-x_A)^2+(y_10-y_A)^2-L_A_10^2;
    r(5)=(x_1-x_2)^2+(y_1-y_2)^2-L_1_2^2;
    r(6)=(x_9-x_10)^2+(y_9-y_10)^2-L_9_10^2;
    r(7)=(x_2-x_10)^2+(y_2-y_10)^2-L_2_10^2;

    r(8)=(x_17-x_A)^2+(y_17-y_A)^2-L_A_17^2;

    r(9)=(x_1-x_18)^2+(y_1-y_18)^2-L_1_18^2;
    r(10)=(x_1-x_18)*L_1_3-(x_1-x_3)*L_1_18;
    r(11)=(y_1-y_18)*L_1_3-(y_1-y_3)*L_1_18;

    r(12)=(x_9-x_19)^2+(y_9-y_19)^2-L_9_19^2;
    r(13)=(x_19-x_9)*L_9_11-(x_11-x_9)*L_9_19;
    r(14)=(y_19-y_9)*L_9_11-(y_11-y_9)*L_9_19;

    r(15)=(x_2-x_4)^2+(y_2-y_4)^2-L_2_4^2;
    r(16)=(x_10-x_12)^2+(y_10-y_12)^2-L_10_12^2;

    r(17)=(x_3-x_5)^2+(y_3-y_5)^2-L_3_5^2;
    r(18)=(x_5-x_3)*L_3_4-(x_4-x_3)*L_3_5;
    r(19)=(y_5-y_3)*L_3_4-(y_4-y_3)*L_3_5;

    r(20)=(x_11-x_13)^2+(y_11-y_13)^2-L_11_13^2;
    r(21)=(x_13-x_11)*L_11_12-(x_12-x_11)*L_11_13;
    r(22)=(y_13-y_11)*L_11_12-(y_12-y_11)*L_11_13;

    r(23)=(x_5-x_7)^2+(y_5-y_7)^2-L_5_7^2;
    r(24)=(x_6-x_7)*(x_5-x_7)+(y_6-y_7)*(y_5-y_7)-L_6_7*L_5_7*cos(pi/2);
    r(25)=(x_3-x_5)*(x_7-x_5)+(y_3-y_5)*(y_7-y_5)-L_3_5*L_5_7*cos(kingpin);

    r(26)=(x_13-x_15)^2+(y_13-y_15)^2-L_13_15^2;
    r(27)=(x_14-x_15)*(x_13-x_15)+(y_14-y_15)*(y_13-y_15)-L_14_15*L_13_15*cos(pi/2);
    r(28)=(x_11-x_13)*(x_15-x_13)+(y_11-y_13)*(y_15-y_13)-L_11_13*L_13_15*cos(kingpin);

    r(29)=(x_6-x_8)^2 + (y_6-y_8)^2-L_6_8^2;
```

```
    r(30)=(x_7-x_6)*L_6_8-(x_8-x_6)*L_6_7;
    r(31)=(y_7-y_6)*L_6_8-(y_8-y_6)*L_6_7;

    r(32)=(x_14-x_16)^2 + (y_14-y_16)^2-L_14_16^2;
    r(33)=(x_15-x_14)*L_14_16-(x_16-x_14)*L_14_15;
    r(34)=(y_15-y_14)*L_14_16-(y_16-y_14)*L_14_15;

    r(35)=(x_C-x_B)*(y_6-y_B)-(y_C-y_B)*(x_6-x_B);
    r(36)=(x_C-x_B)*(y_14-y_B)-(y_C-y_B)*(x_14-x_B);

    r(37)=(x_17-x_18)^2 + (y_17-y_18)^2-s_1^2;
    r(38)=(x_17-x_19)^2 + (y_17-y_19)^2-s_2^2;

    %r(39)=(x_9-x_A)-L_A_9*cos(phi);
    r(39)=(x_9-x_A)*(x_17-x_A)+(y_9-y_A)*(y_17-y_A)-L_A_9*L_A_17*cos(phi);
    %r(40)=s_1-s_10; r(41)=s_2-s_20;
    %r(40)=(y_9-y_A)-L_A_9*sin(phi);
end
```

Reactions

```
function [lambda,T]=Reactions(tiempo,yCompleto)
    global q qini n m

    load('variables.mat','r','r2','J','vars','J2','varst','M','Q');
    numIncr=size(yCompleto,1); %Obtiene el numero de incrementos de tiempo
    lambda=zeros(numIncr,m);  %Inicializa el vector de multip. de Lagrange
    T=zeros(numIncr,1);  %Inicializa el vector de multip. de Lagrange
    [B,I]=sort(vars);
    q = qini;
    for i=1:numIncr
        y=yCompleto(i,:);
        yp = Deriv(tiempo, y);
        ang = y(1);
        angp =y(2);
        angpp=yp(2);

        r=ConstraintEq; r=matlabFunction(r);
        q=PositionProblem(ang,vars,r,J);
        qp=VelocityProblem(q,angp,vars,J);
        qpp=AcelerationProblem(q,qp,angpp,varst,J,J2);

        aux=num2cell(q(I));
        matJacob=J(aux{1:n});
        lambda(i,:)=pinv(matJacob')*(Q-M*qpp);
        T(i)=(matJacob(:,n)')*(lambda(i,:)');
    end
end
```

Kinematic Simulation

```
clear all
clc
close all

InitialData;

phip=1;
phipp=0;

i=1;
figure('Units','normalized','Position',[0 0 1 1]);
for deltaphi=0:0.018:(25*pi/180)

    q=PositionProblem(phi_val+deltaphi,vars,r,J);
    qp=VelocityProblem(q,phip,vars,J);
    %qpp=AcelerationProblem(q,qp,phipp,varst,J,J2);

    cla;
    DrawMechanism(q);
    title(['phi=', num2str(phi_val+deltaphi)]);
    drawnow;

    %Variables a Guardar
%     x_1s(i)=q(1);
    i=i+1;
end

phi_left=kingpin-acos((x_4s-x_3s)/L_3_4); %phi_left(end)=[];
phi_right=kingpin-acos((x_1s-x_2s)/L_1_2); %phi_right(end)=[];

md1 = polyfit(phis*180/pi,phi_left*180/pi,1);
md2 = polyfit(phis*180/pi,phi_right*180/pi,1);

figure('Units','normalized','Position',[0 0 1 1]);
ax1=subplot(1,2,1);
plot(polyval(md1,phis*180/pi)); xlabel('Body_Angle'); ylabel('Left_Wheel_Angle'); title('
    ↪ Body_vs_Left_Wheel_Angle'); hold on
scatter(phis*180/pi,phi_left*180/pi,'r*')
ax2=subplot(1,2,2); ax2.YGrid='on';
plot(polyval(md2,phis*180/pi)); xlabel('Body_Angle'); ylabel('Right_Wheel_Angle'); title('
    ↪ Body_vs_Right_Wheel_Angle'); hold on; set(gca,'YAxisLocation','right');
scatter(phis*180/pi,phi_right*180/pi,'r*')
linkaxes([ax1,ax2],'xy'); ax1.YGrid='on'; ax2.YGrid='on';

phi_max=max(phis);
```

Dynamic Simulation

```
clear all
clc
close all


InitialData;

phi_ini=phi_val;
phip_ini= 0*pi/180;

y0=[phi_ini, phip_ini];

[M, Q] =Calculate_M_Q(phi_ini);

tInicial    = 0;
tFinal      = 0.1;
tIncrement  = 0.1;
tspan       = tInicial:tIncrement:tFinal;

%Par minimo movimiento
q = PositionProblem(phi_ini,vars,r,J);
R=Calculate_R(q,vars,J);
qp=R*phip_ini;
Rpzp= AcelerationProblem(q,qp,0,varst,J,J2);
Q_par=R'*Q-R'*M*Rpzp;
%Q(n)=-Q_par;
%Q(n)=0.01;

save('variables.mat','r','r2','J','vars','J2','varst','M','Q');

options=odeset('AbsTol',1e-0,'RelTol',1e-0,'OutputFcn',@odeplot);
[t,y] = ode45(@Deriv, tspan, y0, options);

DrawAnimation(t,y);

%[lambda,T]=Reactions(t,y);

response=zeros(size(y,1),4);
response(:,1)=y(:,1);
for i=1:size(y,1)
    [ans, response(i,4)]=(Deriv(t(i),y(i,:)));
    response(i,2:3)=ans';
end
```

Position Problem

```
function qnew=PositionProblem(phi_val,vars,r,J)

    global q n

    error=1e+10;
    epsilon=1e-3;
    qnew=q;


    iter=1;
    max_iter=20;
    [B,I]=sort(vars);
    while (error>epsilon  && iter < max_iter)
        qnew(n)=phi_val;

        aux=num2cell(qnew(I));
        vectorEc=r(aux{1:n})';
        mJacob=J(aux{1:n});

        delta_q=-pinv(mJacob)*vectorEc;
        %delta_q=-mJacob\vectorEc;
        qnew=qnew + delta_q;
        error=norm(delta_q);
        iter=iter+1;
    end
    norm(vectorEc);
end
```

Velocity Problem

```
function qp = VelocityProblem(q,phip,vars,J)

    global m n
    [B,I]=sort(vars);
    aux=num2cell(q(I));

    mJacob=zeros(m+1,n);

    mJacob(1:m,:)=J(aux{1:n});
    mJacob(m+1,n)=1;

    b=zeros(m+1,1);
    b(m+1)=phip;

    qp=-mJacob\b;
end
```

Acceleration Problem

```
function qpp = AcelerationProblem(q,qp,phipp,varst,J,J2)

    global m n

    [B,I]=sort(varst);
    aux=num2cell(q(I));

    mJacob=zeros(m+1,n);
    mJacob(1:m,:) = J(aux{1:n});
    mJacob(m+1,n)=1;

    b=zeros(m+1,1);

    Fiqp=double(subs(J2,[varst,diff(varst)],[q,qp]));

    b(1:m)= Fiqp*qp;
    b(m)=phipp;

    qpp=-mJacob\b;
end
```

Calculate R

```
function R = Calculate_R(q,vars,J)

    global m n

    [B,I]=sort(vars);
    aux=num2cell(q(I));

    mJacob=zeros(m+1,n);

    mJacob(1:m,:)=J(aux{1:n});
    mJacob(m+1,n)=1;

    b=zeros(m+1,1);
    b(m+1)=1;

    R=pinv(mJacob)*b;

end
```

Calculate M and Q

```
function [M, Q] =Calculate_M_Q(t)
    load('Constants.mat');
    load('Initial_data.mat');

    M=zeros(n,n);
    M_e=[2 0 1 0;
        0 2 0 1;
        1 0 2 0;
        0 1 0 2;]/6;

    Q=zeros(n,1);
    g=9.81;

    for k=1:2:length(barras)
        %Barra i,j
        i=barras(k);
        j=barras(k+1);
        m_i_j=masa((k+1)/2);

        pos_x_i=find(vars==strcat('x_',num2str(i{1})));
        pos_y_i=find(vars==strcat('y_',num2str(i{1})));
        pos_x_j=find(vars==strcat('x_',num2str(j{1})));
        pos_y_j=find(vars==strcat('y_',num2str(j{1})));

        %len=eval(strcat('L_',num2str(i{1}),'_',num2str(j{1})));
        p=pos((k+1)/2,:);
        F=F_apl(:,(k+1)/2);
        T_p=[1-p(1) p(2) p(1) -p(2);
            -p(2) 1-p(1) p(2) p(1)];
        Q_temp=T_p'*F;

        if(isempty(pos_x_i))
            M(pos_x_j:pos_y_j,pos_x_j:pos_y_j)=M(pos_x_j:pos_y_j,pos_x_j:pos_y_j)+M_e
                ↪ (3:4,3:4)*m_i_j;
            Q(pos_x_j:pos_y_j)=Q(pos_x_j:pos_y_j)+Q_temp(3:4);
            continue
        end
        if(isempty(pos_x_j))
            M(pos_x_i:pos_y_i,pos_x_i:pos_y_i)=M(pos_x_i:pos_y_i,pos_x_i:pos_y_i)+M_e
                ↪ (1:2,1:2)*m_i_j;
            Q(pos_x_i:pos_y_i)=Q(pos_x_i:pos_y_i)+Q_temp(1:2);
            continue
        end
        M(pos_x_i:pos_y_i,pos_x_i:pos_y_i)=M(pos_x_i:pos_y_i,pos_x_i:pos_y_i)+M_e(1:2,1:2)
            ↪ *m_i_j;
        M(pos_x_i:pos_y_i,pos_x_j:pos_y_j)=M(pos_x_i:pos_y_i,pos_x_j:pos_y_j)+M_e(1:2,3:4)
            ↪ *m_i_j;
```

```
        M(pos_x_j:pos_y_j,pos_x_i:pos_y_i)=M(pos_x_j:pos_y_j,pos_x_i:pos_y_i)+M_e(3:4,1:2)
            ↪ *m_i_j;
        M(pos_x_j:pos_y_j,pos_x_j:pos_y_j)=M(pos_x_j:pos_y_j,pos_x_j:pos_y_j)+M_e(3:4,3:4)
            ↪ *m_i_j;

        Q(pos_x_i:pos_y_i)=Q(pos_x_i:pos_y_i)+Q_temp(1:2);
        Q(pos_x_j:pos_y_j)=Q(pos_x_j:pos_y_j)+Q_temp(3:4);
    end
    Q(n)=0; R=10; V=8; m_total=130;
    F_y=(m_total*V^2/R)/3;
    F_y=100;
    Q(11)=Q(11)-F_y; Q(27)=Q(27)-F_y;

end
```

Deriv Function

```
function [yp,Q_par] = Deriv(t,y)
        load('variables.mat','r','r2','J','vars','J2','varst','M','Q')
        load('Constants.mat');

        z = y(1);
        zp = y(2);
        r=matlabFunction(r2);
        q = PositionProblem(z,vars,r,J);
        R=Calculate_R(q,vars,J);

        qp=R*zp;
        Rpzp= AcelerationProblem(q,qp,0,varst,J,J2);

        K=1000;
        alpha_1=abs(atan((q(36)-q(34))/(q(35)-q(33))));
        F_x= K*(q(39)-s_10)*cos(alpha_1);
        F_y= K*(q(39)-s_10)*sin(alpha_1);
        Q(33)=Q(33)-F_x; Q(34)=Q(34)+F_y;
        Q(35)=Q(35)+F_x; Q(36)=Q(36)-F_y;

        alpha_2=abs(atan((q(38)-q(34))/(q(37)-q(33))));
        F_x= K*(q(40)-s_20)*cos(alpha_2);
        F_y= K*(q(40)-s_20)*sin(alpha_2);
        Q(33)=Q(33)+F_x; Q(34)=Q(34)+F_y;
        Q(37)=Q(35)-F_x; Q(38)=Q(36)-F_y;

        Q_par=R'*Q-R'*M*Rpzp;
        %Q(n)=-2*Q_par;
        zpp=(R'*M*R)\(R'*(Q-M*Rpzp));
        yp=[zp; zpp];
end
```

Draw Mechanism

```
function DrawMechanism(q)

    load('Constants.mat');

    x_1=q(1); y_1=q(2);    x_2=q(3); y_2=q(4);    x_3=q(5); y_3=q(6);    x_4=q(7); y_4=q(8);
    x_5=q(9); y_5=q(10);    x_6=q(11); y_6=q(12);    x_7=q(13); y_7=q(14);    x_8=q(15); y_8=q(16);
    x_9=q(17); y_9=q(18);    x_10=q(19); y_10=q(20);    x_11=q(21); y_11=q(22);    x_12=q(23); y_12=q(24);
    x_13=q(25); y_13=q(26);    x_14=q(27); y_14=q(28);    x_15=q(29); y_15=q(30);    x_16=q(31); y_16=q(32);
    x_17=q(33); y_17=q(34);    x_18=q(35); y_18=q(36);    x_19=q(37); y_19=q(38);    s_1=q(39); s_2=q(40);
    phi=q(41);

%     phi_aux=phi-atan(L_1_9/(2*h));
%     h_g=L_6_7+L_3_5*sin(kingpin)+h;

    lineWidth=3;
    color_3=[74, 160, 213]/255;
    color_2=[74, 160, 213]/255;
    color_1=[250, 181, 122]/255;
    color_4=[121, 168, 169]/255;

    hold on
    axis equal;
    axis off;

    set(gca,'Color',[0 0 0]);
    fill([x_1, x_9, x_10, x_2], [y_1, y_9, y_10, y_2], color_2,'EdgeColor','None');

    %line([x_A, x_A+h_g*sin(phi_aux)],[y_A, y_A-h_g*cos(phi_aux)],'lineWidth',1,'lineStyle','--','color',color_2)
%    line([x_A, x_A],[y_A, y_A-h_g],'lineWidth',1,'lineStyle','--','color',color_2)

    line([x_B, x_C],[y_B, y_C], 'lineWidth',1,'color',color_1);
    line([x_5, x_7],[y_5, y_7], 'lineWidth',lineWidth,'color',color_1);
    line([x_3, x_5],[y_3, y_5], 'lineWidth',lineWidth,'color',color_1);
    line([x_4, x_5],[y_4, y_5], 'lineWidth',lineWidth,'color',color_1);
    line([x_1, x_3],[y_1, y_3], 'lineWidth',lineWidth,'color',color_1);
    line([x_2, x_4],[y_2, y_4], 'lineWidth',lineWidth,'color',color_1);
    line([x_1, x_2],[y_1, y_2], 'lineWidth',lineWidth,'color',color_1);
    %line([x_1, x_A],[y_1, y_A], 'lineWidth',lineWidth,'color',color_1);
    %line([x_2, x_A],[y_2, y_A], 'lineWidth',lineWidth,'color',color_1);
    line([x_6, x_7],[y_6, y_7], 'lineWidth',lineWidth*2,'color',color_2);
    line([x_7, x_8],[y_7, y_8], 'lineWidth',lineWidth*2,'color',color_2);
```

```
42
43          line([x_13, x_15],[y_13, y_15], 'lineWidth',lineWidth,'color',color_1);
44      line([x_12, x_13],[y_12, y_13], 'lineWidth',lineWidth,'color',color_1);
45      line([x_11, x_13],[y_11, y_13], 'lineWidth',lineWidth,'color',color_1);
46      line([x_9, x_11],[y_9, y_11], 'lineWidth',lineWidth,'color',color_1);
47      line([x_10, x_12],[y_10, y_12], 'lineWidth',lineWidth,'color',color_1);
48      line([x_9, x_10],[y_9, y_10], 'lineWidth',lineWidth,'color',color_1);
49      line([x_9, x_A],[y_9, y_A], 'lineWidth',lineWidth,'color',color_1);
50      line([x_1, x_A],[y_1, y_A], 'lineWidth',lineWidth,'color',color_1);
51      line([x_14, x_15],[y_14, y_15], 'lineWidth',lineWidth*2,'color',color_2);
52      line([x_15, x_16],[y_15, y_16], 'lineWidth',lineWidth*2,'color',color_2);
53      line([x_17, x_A],[y_17, y_A], 'lineWidth',lineWidth,'color',color_1);
54
55      line([x_1, x_9],[y_1, y_9], 'lineWidth',lineWidth,'color',color_1);
56      line([x_2, x_10],[y_2, y_10], 'lineWidth',lineWidth,'color',color_1);
57
58      r=5e-3;
59      t=linspace(0,2*pi,100);
60      fill(x_A+r*cos(t),y_A+r*sin(t),color_3,'EdgeColor','None');
61      fill(x_1+r*cos(t),y_1+r*sin(t),color_3,'EdgeColor','None');
62      fill(x_2+r*cos(t),y_2+r*sin(t),color_3,'EdgeColor','None');
63      fill(x_3+r*cos(t),y_3+r*sin(t),color_3,'EdgeColor','None');
64      fill(x_4+r*cos(t),y_4+r*sin(t),color_3,'EdgeColor','None');
65      fill(x_9+r*cos(t),y_9+r*sin(t),color_3,'EdgeColor','None');
66      fill(x_10+r*cos(t),y_10+r*sin(t),color_3,'EdgeColor','None');
67      fill(x_11+r*cos(t),y_11+r*sin(t),color_3,'EdgeColor','None');
68      fill(x_12+r*cos(t),y_12+r*sin(t),color_3,'EdgeColor','None');
69      fill(x_17+r*cos(t),y_17+r*sin(t),color_3,'EdgeColor','None');
70
71      h_box=10e-3; b_box=20e-3;
72      rectangle('Position',[x_6-b_box/2 y_6-h_box/2 b_box h_box],'Curvature',0.4,'EdgeColor'
            ↪ ,color_1);
73      rectangle('Position',[x_14-b_box/2 y_14-h_box/2 b_box h_box],'Curvature',0.4,'
            ↪ EdgeColor',color_1);
74
75      %Muelles;
76      pA=[x_18 y_18];
77      pB=[x_17 y_17];
78      x_C=pA(1)+0.4*(pB(1)-pA(1)); y_C=pA(2)+0.4*(pB(2)-pA(2));
79      x_F=pA(1)+0.6*(pB(1)-pA(1)); y_F=pA(2)+0.6*(pB(2)-pA(2));
80      x_D=x_C-s_1*(pB(2)-pA(2)); y_D=y_C+s_1*(pB(1)-pA(1));
81      x_E=x_F+s_1*(pB(2)-pA(2)); y_E=y_F-s_1*(pB(1)-pA(1));
82      line([pA(1), x_C],[pA(2), y_C], 'lineWidth',lineWidth/2,'color',color_2);
83      line([pB(1), x_F],[pB(2), y_F], 'lineWidth',lineWidth/2,'color',color_2);
84      line([x_C, x_D],[y_C, y_D], 'lineWidth',lineWidth/2,'color',color_2);
85      line([x_E, x_D],[y_E, y_D], 'lineWidth',lineWidth/2,'color',color_2);
86      line([x_E, x_F],[y_E, y_F], 'lineWidth',lineWidth/2,'color',color_2);
87
88      pA=[x_19 y_19];
```

```
    pB=[x_17 y_17];
    x_C=pA(1)+0.4*(pB(1)-pA(1)); y_C=pA(2)+0.4*(pB(2)-pA(2));
    x_F=pA(1)+0.6*(pB(1)-pA(1)); y_F=pA(2)+0.6*(pB(2)-pA(2));
    x_D=x_C-s_1*(pB(2)-pA(2)); y_D=y_C+s_1*(pB(1)-pA(1));
    x_E=x_F+s_1*(pB(2)-pA(2)); y_E=y_F-s_1*(pB(1)-pA(1));
    line([pA(1), x_C],[pA(2), y_C], 'lineWidth',lineWidth/2,'color',color_2);
    line([pB(1), x_F],[pB(2), y_F], 'lineWidth',lineWidth/2,'color',color_2);
    line([x_C, x_D],[y_C, y_D], 'lineWidth',lineWidth/2,'color',color_2);
    line([x_E, x_D],[y_E, y_D], 'lineWidth',lineWidth/2,'color',color_2);
    line([x_E, x_F],[y_E, y_F], 'lineWidth',lineWidth/2,'color',color_2);

    set(gcf, 'color', [255 255 255]/256)
    drawnow;
end
```

Draw Animation

```
function DrawAnimation(t, y)

    load('Initial_data.mat');
    global qini
    q = qini;

    figure('Units','normalized','Position',[0 0 1 1]);
    numIncr=size(y,1);

    for i=1:numIncr,
        cla;
        phi_val = y(i,1);
        r=ConstraintEq;
        %r(10)=r(10)-phi_val;
        r=matlabFunction(r);
        q = PositionProblem(phi_val,vars,r,J);
        DrawMechanism(q);
        drawnow;
        title(['t=', num2str(t(i))]);
        pause(0.01);
    end

    figure('Units','normalized','Position',[0 0 1 1]);
    plot(t,y);
    hold on;
    legend('phi', 'phi_p');
end
```

Model Gains

```
%% Controller with varying longitudinal velocity
clear all
clc
close all

v=[1:18]';
inverse=1; %boolean activating speed-inverse component
if(inverse)
    M=[ones(length(v),1) v 1./v];
else
    M=[ones(length(v),1) v];
end
R_v=zeros(length(v),7);

for (i=1:length(v))
    R_v(i,:)=get_Gains(v(i));
end
reg=(inv(M'*M)*M'*R_v);
K_c=reg(1,:);
K_v=reg(2,:);
if(inverse)
    K_1_v=reg(3,:);
else
    K_1_v=zeros(1,7);
end

figure('units','normalized','outerposition',[0 0 1 1]);
set(gcf,'color','w');set(0,'defaultAxesFontName', 'Montserrat');set(0,'defaultTextFontName
    ↪ ', 'Montserrat');
var={'a_{per}';'\Psi^\prime';'\theta';'\theta^\prime';'a_{per}^I';'\delta';'\delta^\prime'
    ↪ };
for (i=1:7)
   subplot(2,4,i);
   x=[v(1):0.02:v(end)];   y=K_c(i)+K_v(i)*x +K_1_v(i)./x;
   plot(x,y,'Color',[216, 71, 151]./255,'LineWidth',1.5);  hold on
   y_v=K_c(i)+K_v(i)*v + K_1_v(i)./v;
   plot(v,y_v,'o','MarkerSize',3.0,'Color',[130, 9, 51]./255);

   title(strcat(var(i),'_gain'),'FontWeight','normal');
   xlabel('V_{x}_(m/s)');  xlim([v(1) v(end)]);
   ax=gca;    ax.XGrid = 'on'; ax.XTick=[2:2:18];
end
h=legend('Gain_function_of_V_x','Calculated_Gains');
set(h,'Position',[0.75,0.37 0.110, 0.074]);
```

getGains Function

```
function [Gains] = get_Gains(V)
%% Parameters

%PEV Values with driver
% L_f=0.71;
% L_r=0.61;
% h=0.91;
% m=115;
% I_x=25;
% I_z=18;
% C_f=7277;
% C_r=20500;
% landa_f=230;
% landa_r=730;
% g=9.81;

%PEV Values without driver
L_f=0.51;
L_r=0.81;
h=0.36;
m=35;
I_x=4;
I_z=11;
C_f=2700;
C_r=3500;
landa_f=80;
landa_r=105;
g=9.81;

%% Plant Model
a=2*C_f+C_r;
b=2*L_f*C_f-L_r*C_r;
A=[-a*((1/m)+((h^2)/I_x))/V (-b*((1/m)+((h^2)/I_x))/V)-V ((2*landa_f+landa_r)*((1/m)+((h
    ↪ ^2)/I_x)))-((m*g*h^2)/I_x) 0
    -b/(V*I_z) -(2*C_f*L_f^2+C_r*L_r^2)/(V*I_z) (2*landa_f*L_f-landa_r*L_r)/I_z 0
    0 0 0 1
    (a*h)/(V*I_x) (b*h)/(V*I_x) (m*g*h-h*(2*landa_f+landa_r))/I_x 0];

B_d=[2*C_f*((1/m)+((h^2)/I_x)) 2*C_f*L_f/I_z 0 -2*C_f*h/I_x]';
B_u=[-h/I_x 0 0 1/I_x]';

%% Perceived acceleration formula
G_1=[0 V -g 0];
G_2=[1 0 0 h];
G=G_1+G_2*A;
H_u=G_2*B_u;
H_d=G_2*B_d;
```

```
47
48 %% Extended state (with a_per)
49 A_i=[A zeros(4,1);G zeros(1,1)];
50 B_iu=[B_u;H_u];
51 B_id=[B_d;H_d];
52 G_i=[0 0 0 0 1];
53 H_iu=0;
54 H_id=0;
55
56 %% Exogenous signal model (steering)
57 alpha=1;
58 beta=0.3;
59 tao=0.5;
60 A_e=[0 1; -tao*alpha -(tao+alpha)];
61 B_e=[0;beta];
62 C_e=[1 0];
63
64 %% Baseline LQR
65 Q=0.1; R=1;
66
67 % Plant Model
68 Q_x=G'*Q*G;
69 R_u=H_u'*Q*H_u + R;
70 N_xu=G'*Q*H_u;
71 K_lqr1=lqr(A,B_u,Q_x,R_u,N_xu);
72
73 [M_1,L,K] = care(A,B_u,Q_x,R_u,zeros(4,1),eye(4));
74 M_2 = sylvester((A'-M_1*B_u*inv(R_u)*B_u'),A_e,M_1*B_d*C_e);
75 K_1=inv(R_u)*B_u'*M_1;
76 K_2=inv(R_u)*B_u'*M_2;
77
78 % Extended Plant Model
79 Q_x=G_i'*Q*G_i;
80 R_u=H_iu'*Q*H_iu + R;
81 N_xu=G_i'*Q*H_iu;
82 K_lqr=lqr(A_i,B_iu,Q_x,R_u,N_xu); % Full state feedback using LQR
83
84 %% Optimal control: Riccati + Slyvester
85
86 [M_1,L,K] = care(A_i,B_iu,Q_x,R_u,zeros(5,1),eye(5)); % Riccati equation from optimal
       ↪ control notes
87 M_2 = sylvester((A_i'-M_1*B_iu*inv(R_u)*B_iu'),A_e,M_1*B_id*C_e); % Sylvester equation
       ↪ from optimal control notes
88 K_1=inv(R_u)*B_iu'*M_1; % Feedback gain
89 K_2=inv(R_u)*B_iu'*M_2; % Feedforward gain
90
91 %% Solving Sylvester Equations Manually
92 A_model=[-A_i(1,1)+A_e(1,1) -A_i(1,2) -A_i(1,3) -A_i(1,4) -A_i(1,5) A_e(2,1) 0 0 0 0 B_iu
       ↪ (1) 0
```

```
    A_e(1,2) 0 0 0 0 -A_i(1,1)+A_e(2,2) -A_i(1,2) -A_i(1,3) -A_i(1,4) -A_i(1,5) 0 B_iu(1)
    -A_i(2,1) -A_i(2,2)+A_e(1,1) -A_i(2,3) -A_i(2,4) -A_i(2,5) 0 A_e(2,1) 0 0 0 B_iu(2) 0
    0 A_e(1,2) 0 0 0 -A_i(2,1) -A_i(2,2)+A_e(2,2) -A_i(2,3) -A_i(2,4) -A_i(2,5) 0 B_iu(2)
    -A_i(3,1) -A_i(3,2) -A_i(3,3)+A_e(1,1) -A_i(3,4) -A_i(3,5) 0 0 A_e(2,1) 0 0 B_iu(3) 0
    0 0 A_e(1,2) 0 0 -A_i(3,1) -A_i(3,2) -A_i(3,3)+A_e(2,2) -A_i(3,4) -A_i(3,5) 0 B_iu(3)
    -A_i(4,1) -A_i(4,2) -A_i(4,3) -A_i(4,4)+A_e(1,1) -A_i(4,5) 0 0 0 A_e(2,1) 0 B_iu(4) 0
    0 0 0 A_e(1,2) 0 -A_i(4,1) -A_i(4,2) -A_i(4,3) -A_i(4,4)+A_e(2,2) -A_i(4,5) 0 B_iu(4)
    -A_i(5,1) -A_i(5,2) -A_i(5,3) -A_i(5,4) -A_i(5,5)+A_e(1,1) 0 0 0 0 A_e(2,1) B_iu(5) 0
    0 0 0 0 A_e(1,2) -A_i(5,1) -A_i(5,2) -A_i(5,3) -A_i(5,4) -A_i(5,5)+A_e(2,2) 0 B_iu(5)
    0 0 0 0 1 0 0 0 0 0 0 0
    0 0 0 0 0 0 0 0 0 1 0 0];
bb=B_id*C_e;
b_model=[bb(1,1) bb(1,2) bb(2,1) bb(2,2) bb(3,1) bb(3,2) bb(4,1) bb(4,2) bb(5,1) bb(5,2) 0
    ↪  0]';

x=pinv(A_model)*b_model; %x=b_model\A_model;
T_a=zeros(5,2); F_a=zeros(1,2);
T_a(1,1)=x(1); T_a(2,1)=x(2);T_a(3,1)=x(3); T_a(4,1)=x(4); T_a(5,1)=x(5);
T_a(1,2)=x(6); T_a(2,2)=x(7);T_a(3,2)=x(8); T_a(4,2)=x(9); T_a(5,2)=x(10);
F_a(1,1)=x(11); F_a(1,2)=x(12);
K_d=F_a+K_1*T_a; % Feedforward feedback

%% Standard Model: H2 optimal control
A_s=[A_i B_id*C_e; zeros(2,5) A_e];
B_s=[B_iu;zeros(2,1)];
Q_01=[G_i 0 0]'*Q*[G_i 0 0];
Q_02=zeros(7,1);
Q_03=R;
[S,L,K_s]=care(A_s,B_s,Q_01,Q_03,Q_02,eye(7));
%%K_h2=inv(Q_03)*(B_s'*S+Q_02');
K_M_t=-K_s(1)*(B_u(1)+h*B_u(4))/(A(1,1)+h*A(4,1));

%% Final Gains

K_x=K_s(1:5); K_xw=K_s(6:7);
%K_x=K_1; K_xw=K_2;
den=A(1,1)+h*A(4,1)-K_x(1)*(B_u(1)+h*B_u(4));
K_a_per=K_x(1)/den;
K_phi_p=  ( K_x(2)*( A(1,1)+h*A(4,1) ) - K_x(1)*( A(1,2)+h*A(4,2)+V ) )/den;
K_theta=  ( K_x(3)*( A(1,1)+h*A(4,1) ) - K_x(1)*( A(1,3)+h*A(4,3)-g ) )/den;
K_theta_p=( K_x(4)*( A(1,1)+h*A(4,1) ) - K_x(1)*( A(1,4)+h*A(4,4)   ) )/den;
K_a_per_i=( K_x(5)*( A(1,1)+h*A(4,1) ) )/den;
K_delta=  (K_xw(1)*( A(1,1)+h*A(4,1) ) - K_x(1)*( B_d(1)+h*B_d(4)   ) )/den;
K_delta_p=(K_xw(2)*( A(1,1)+h*A(4,1) ) )/den;
Gains=[K_a_per K_phi_p K_theta K_theta_p K_a_per_i K_delta K_delta_p];

end
```

Model Analysis

```
1
2  %% Load Model
3  load('Modelo_no_driver.mat');
4  % load('Modelo_yes_driver.mat');
5
6  %% State space model
7  t = 0:0.01:10;
8  u = zeros(size(t));
9  x0 = [0 0 0 0 0 0 0.5];
10
11 % Open-loop model
12 states = {'a_{per}';'\Psi^\prime';'\theta';'\theta^\prime';'a_{per}^I';'\delta';'\delta^\
       ↪ prime'};
13 inputs = {'M_{t}'};
14 outputs = {'a_{per}';'\Psi^\prime';'\theta';'\theta^\prime';'a_{per}^I';'\delta';'\delta^\
       ↪ prime'};
15 sys = ss(A_s,B_s,eye(7),0,'statename',states,'inputname',inputs,'outputname',outputs);
16
17 poles_open = eig(A_s); %eigenvalues = closed loop poles
18 cont_rank = rank(ctrb(sys)) %controllability matrix
19 obs_rank = rank(obsv(sys)) %observability matrix
20 pzmap(zpk(sys)); grid; %Pole-zero map
21 iopzplot(sys(1:5)); grid; %Pole-zero map
22 sigma(sys); grid;
23
24 [y_o,t_o,x_o] = lsim(sys,u,t,x0);
25 plot(t_o,y_o)
26
27 % Close-loop model
28 sys_cl = ss(A_s-B_s*K_s,B_s,eye(7),0,'statename',states,'inputname',inputs,'outputname',
       ↪ outputs);
29 [y_c,t_c,x_c] = lsim(sys_cl,u,t,x0);
30 plot(t_c,y_c)
31
32 transfer=tf(sys_cl); %transfer functions
33 zpk(sys_cl);
34 poles_close=eig(A_s-B_s*K_s); %eigenvalues = closed loop poles
35 pzmap(zpk(sys_cl)); grid; %Pole-zero map
36 iopzplot(sys_cl(1:5)); grid; %Pole-zero map
37 sigma(sys_cl); grid; %Singular value plot, shows the principal gains of the frequency
       ↪ response
38 %rlocus(sys_cl(1))
39
40 % vel=2:18;
41 % for i=1:length(vel)
42 %     norm_sys(i)=get_Norm(vel(i));
43 % end
```

```
% plot(norm_sys)

G_plant=tf(sys);
C=ss(K_s);
loopview(G_plant,C);

margins=loopmargin(sys,K_s)
[wcmargi,wcmargo] = wcmargin(sys,K_s)

CL=feedback(G_plant,C)

[stabmarg,destabunc,Report] = robuststab(sys_cl)
[perfmarg,perfmargunc,Report] = robustperf(sys_cl)
```

Model Analysis with Uncertainty

```
%% Parameters

%PEV Values with driver
% L_f=0.71;
% L_r=0.61;
% h=0.91;
% m=115;
% I_z=18;
% I_x=25;
% C_f=3500;
% C_r=3000;
% landa_f=40;
% landa_r=30;
% g=9.81;

%PEV Values without driver
L_f=ureal('L_f',0.51,'Percentage',[-5 5]);
L_r=ureal('L_r',0.81,'Percentage',[-5 5]);
h=ureal('h',0.36,'Percentage',[-15 15]);
m=ureal('m',35,'Percentage',[-20 20]);
I_z=ureal('I_z',11,'Percentage',[-50 50]);
I_x=ureal('I_x',4,'Percentage',[-50 50]);
C_f=ureal('C_f',3500,'Percentage',[-25 25]);
C_r=ureal('C_r',3000,'Percentage',[-25 25]);
landa_f=ureal('landa_f',200,'Percentage',[-25 25]);
landa_r=ureal('landa_r',200,'Percentage',[-25 25]);
g=9.81;

V=7;

I_wheel_f_theta=0.05;
I_wheel_f_rot=0.11;
I_wheel_f_phi=0.05;
```

```
I_wheel_r_theta=0.16;
I_wheel_r_rot=0.32;
I_wheel_r_phi=0.16;
R_wf=0.218; w_rot_f=V/R_wf;
R_wr=0.315; w_rot_r=V/R_wr;
G_phi=-2*(I_wheel_r_theta-I_wheel_r_rot)*w_rot_r/2;
G_theta=2*(I_wheel_f_phi-I_wheel_f_rot)*w_rot_f;

%% Plant Model

a=2*C_f+C_r;
b=2*L_f*C_f-L_r*C_r;
A=[-a*((1/m)+((h^2)/I_x))/V (-b*((1/m)+((h^2)/I_x))/V)-V ((2*landa_f+landa_r)*((1/m)+((h
    ↪ ^2)/I_x)))-((m*g*h^2)/I_x) 0
    -b/(V*I_z) -(2*C_f*L_f^2+C_r*L_r^2)/(V*I_z) (2*landa_f*L_f-landa_r*L_r)/I_z 0
    0 0 0 1
    (a*h)/(V*I_x) (b*h)/(V*I_x) (m*g*h-h*(2*landa_f+landa_r))/I_x 0];

B_d=[2*C_f*((1/m)+((h^2)/I_x)) 2*C_f*L_f/I_z 0 -2*C_f*h/I_x]';
B_u=[-h/I_x 0 0 1/I_x]';

%% Perceived acceleration formula

G_1=[0 V -g 0];
G_2=[1 0 0 h];
G=G_1+G_2*A;
H_u=G_2*B_u;
H_d=G_2*B_d;

%% Extended state (with a_per)

A_i=[A zeros(4,1);G zeros(1,1)];
B_iu=[B_u;H_u];
B_id=[B_d;H_d];

%% Exogenous signal model (steering)

alpha=1;
beta=0.3;
tao=0.5;
A_e=[0 1; -tao*alpha -(tao+alpha)];
B_e=[0;beta];
C_e=[1 0];

%% Standard Model: H2 optimal control
A_s=[A_i B_id*C_e; zeros(2,5) A_e];
B_s=[B_iu;zeros(2,1)];

%% Load Feedback Gains
```

```
load('Modelo_no_driver.mat','K_s');
% load('Modelo_yes_driver.mat','K_s');

%% State space model
t = 0:0.01:10;
u = zeros(size(t));
x0 = [0 0 0 0 0 0 0.5];

% Open-loop model
states = {'a_{per}';'\Psi^\prime';'\theta';'\theta^\prime';'a_{per}^I';'\delta';'\delta^\
    ↪ prime'};
inputs = {'M_{t}'};
outputs = {'a_{per}';'\Psi^\prime';'\theta';'\theta^\prime';'a_{per}^I';'\delta';'\delta^\
    ↪ prime'};
sys = ss(A_s,B_s,eye(7),0,'statename',states,'inputname',inputs,'outputname',outputs);

pzmap(zpk(sys)); grid; %Pole-zero map
iopzplot(sys); grid; %Pole-zero map
sigma(sys); grid;

[y_o,t_o,x_o] = lsim(sys,u,t,x0);
plot(t_o,y_o)

% Close-loop model
sys_cl = ss(A_s-B_s*K_s,B_s,eye(7),0,'statename',states,'inputname',inputs,'outputname',
    ↪ outputs);
[y_c,t_c,x_c] = lsim(sys_cl,u,t,x0);
plot(t_c,y_c)

pzmap(zpk(sys_cl)); grid; %Pole-zero map
iopzplot(sys_cl); grid; %Pole-zero map
sigma(sys_cl); grid; %Singular value plot, shows the principal gains of the frequency
    ↪ response
%rlocus(sys_cl(1))

G_plant=tf(sys);
C=ss(K_s);
loopview(G_plant,C);

margins=loopmargin(sys,K_s)

CL=feedback(G_plant,C);

[stabmarg,destabunc,Report] = robuststab(sys_cl)
[perfmarg,perfmargunc,Report] = robustperf(sys_cl)
```

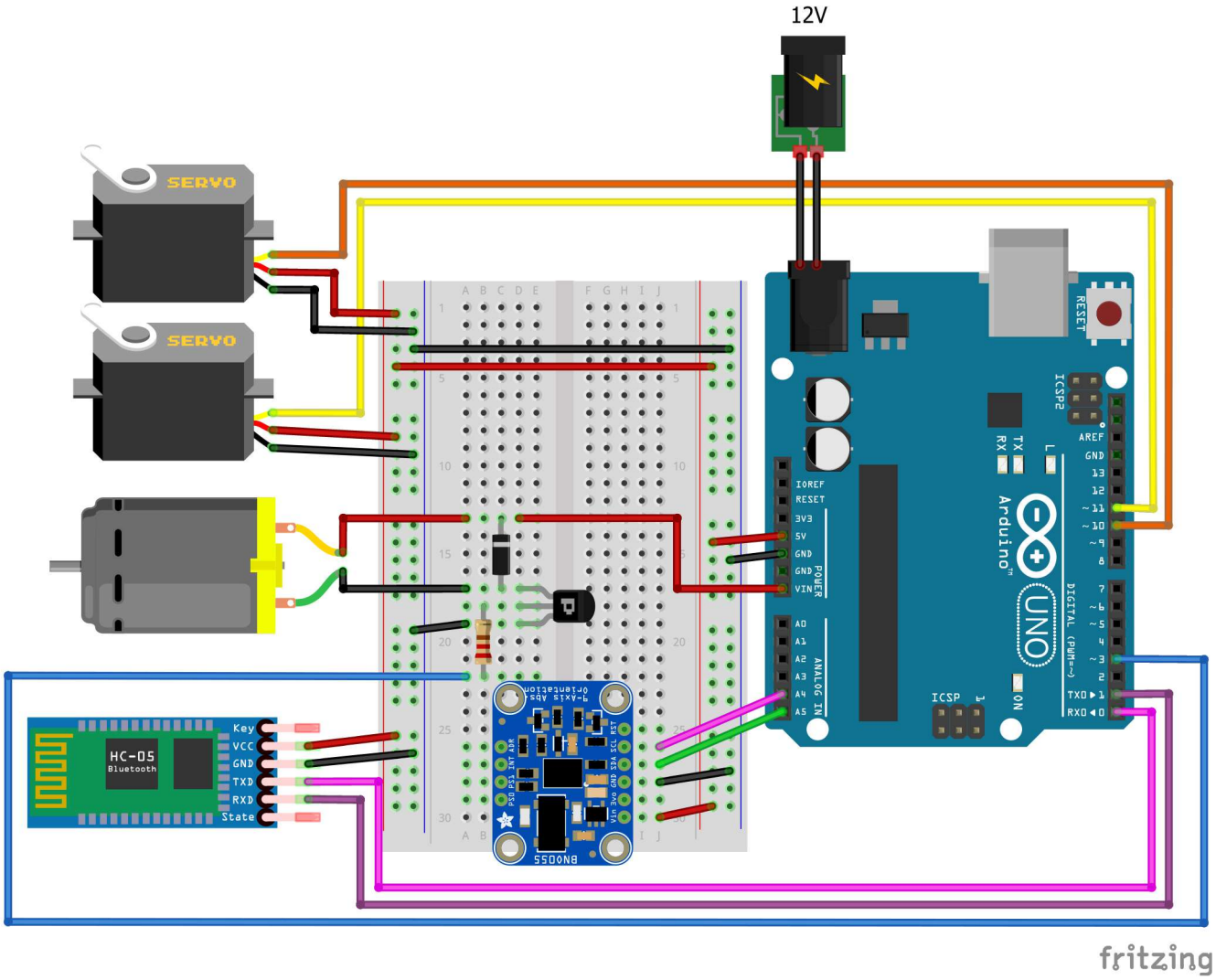
Schematics

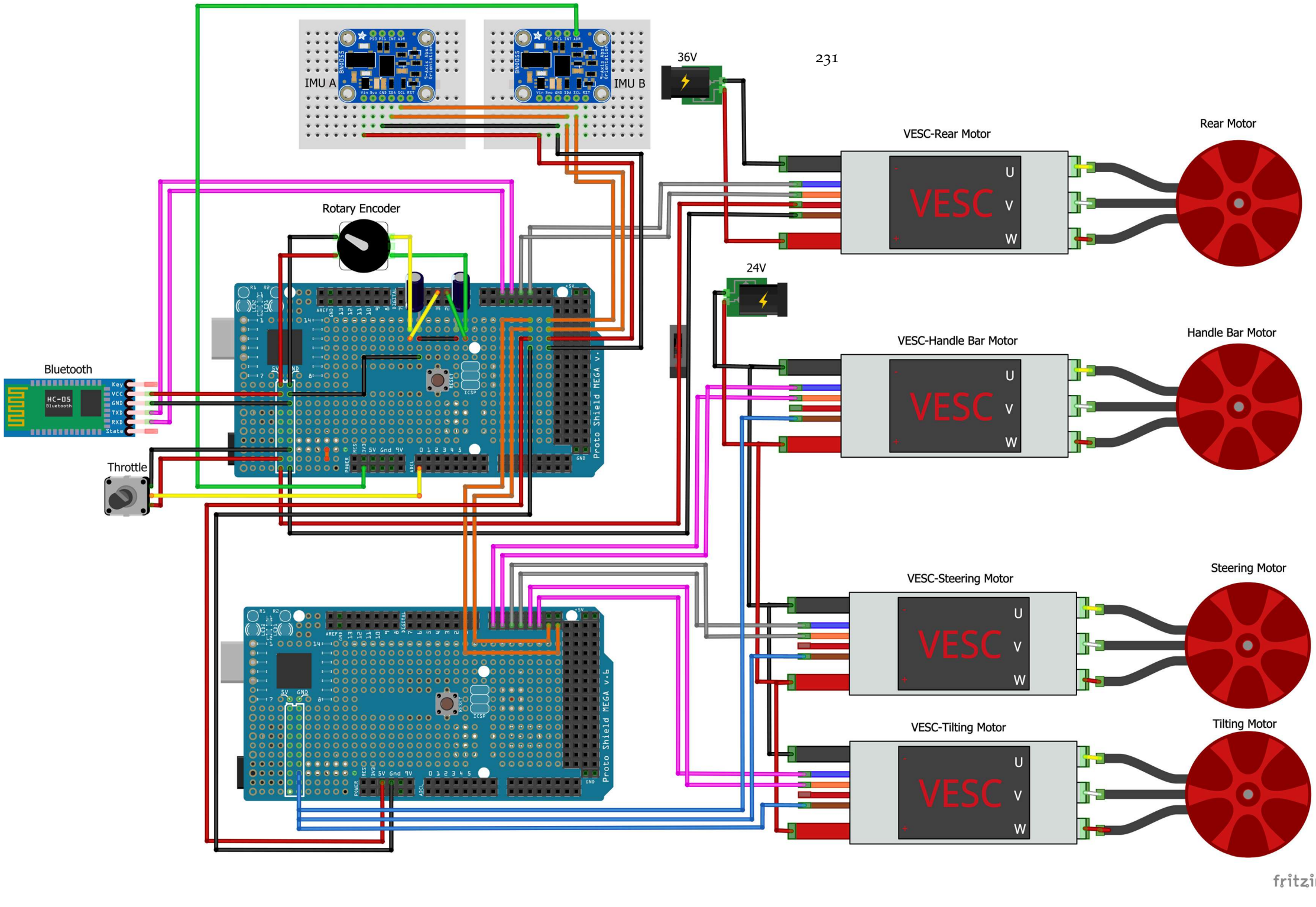
miniPEV Electronics

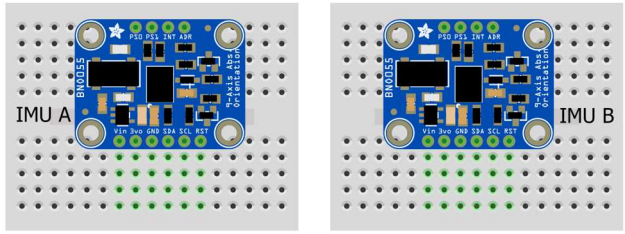
PEV Electronics

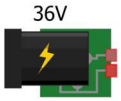
PEV Electronics (without wiring)

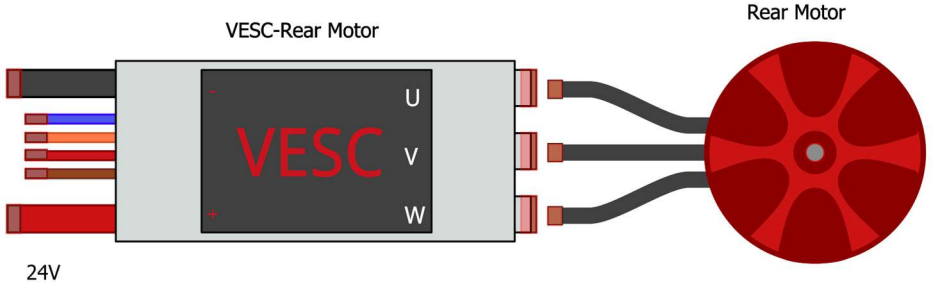
BLDC Motor Controller


12V
SERVO
SERVO
HC-05
Bluetooth
Key
VCC
GND
TXD
RXD
State
BNO055
Arduino
UNO
fritzing

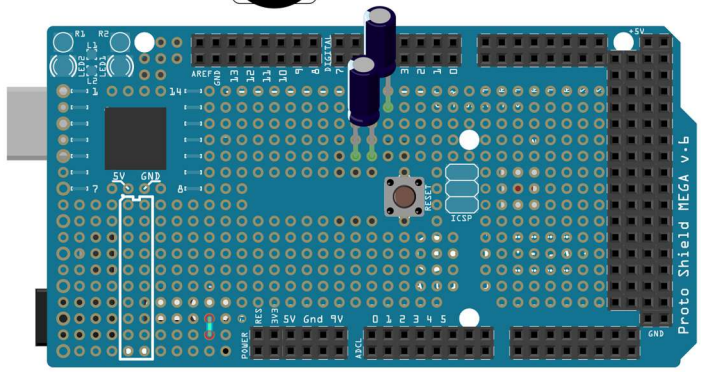

IMU A
IMU B
36V
24V
Rotary Encoder
Bluetooth
HC-05
Bluetooth
Key
VCC
GND
TXD
RXD
State
Throttle
Proto Shield MEGA v.
Proto Shield MEGA v.b
VESC-Rear Motor
VESC
U
V
W
Rear Motor
VESC-Handle Bar Motor
Handle Bar Motor
VESC-Steering Motor
Steering Motor
VESC-Tilting Motor
Tilting Motor
fritzir

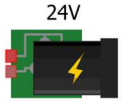


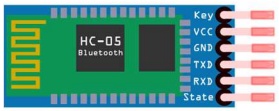


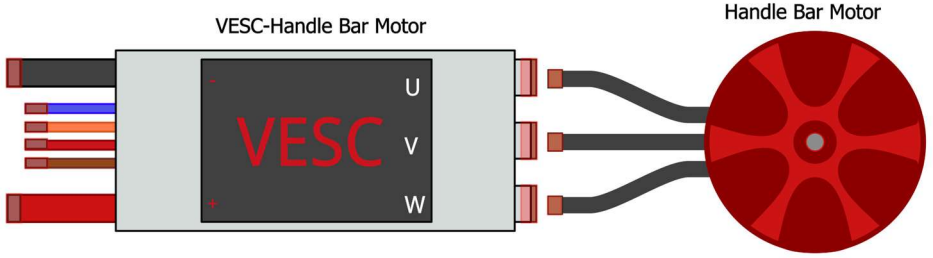


Rotary Encoder

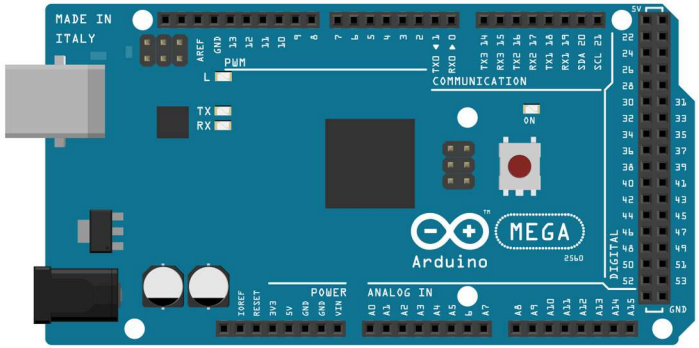


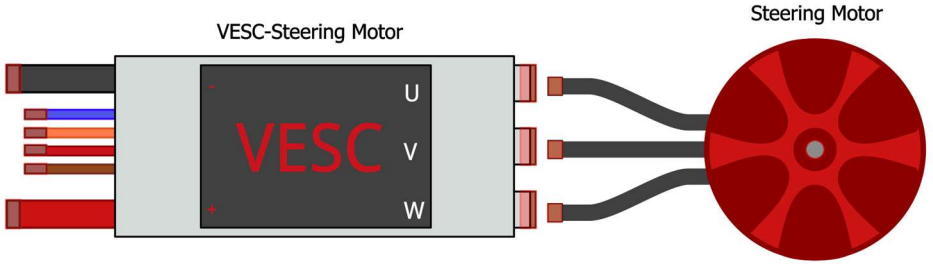


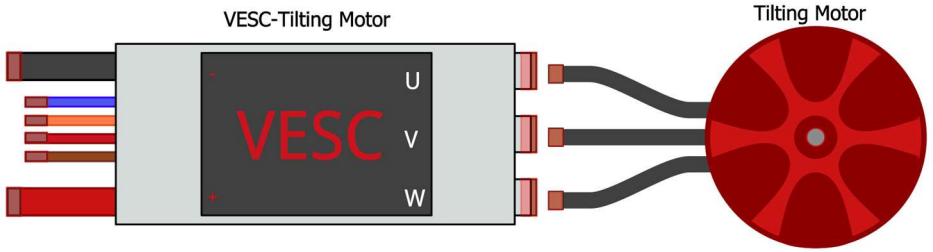




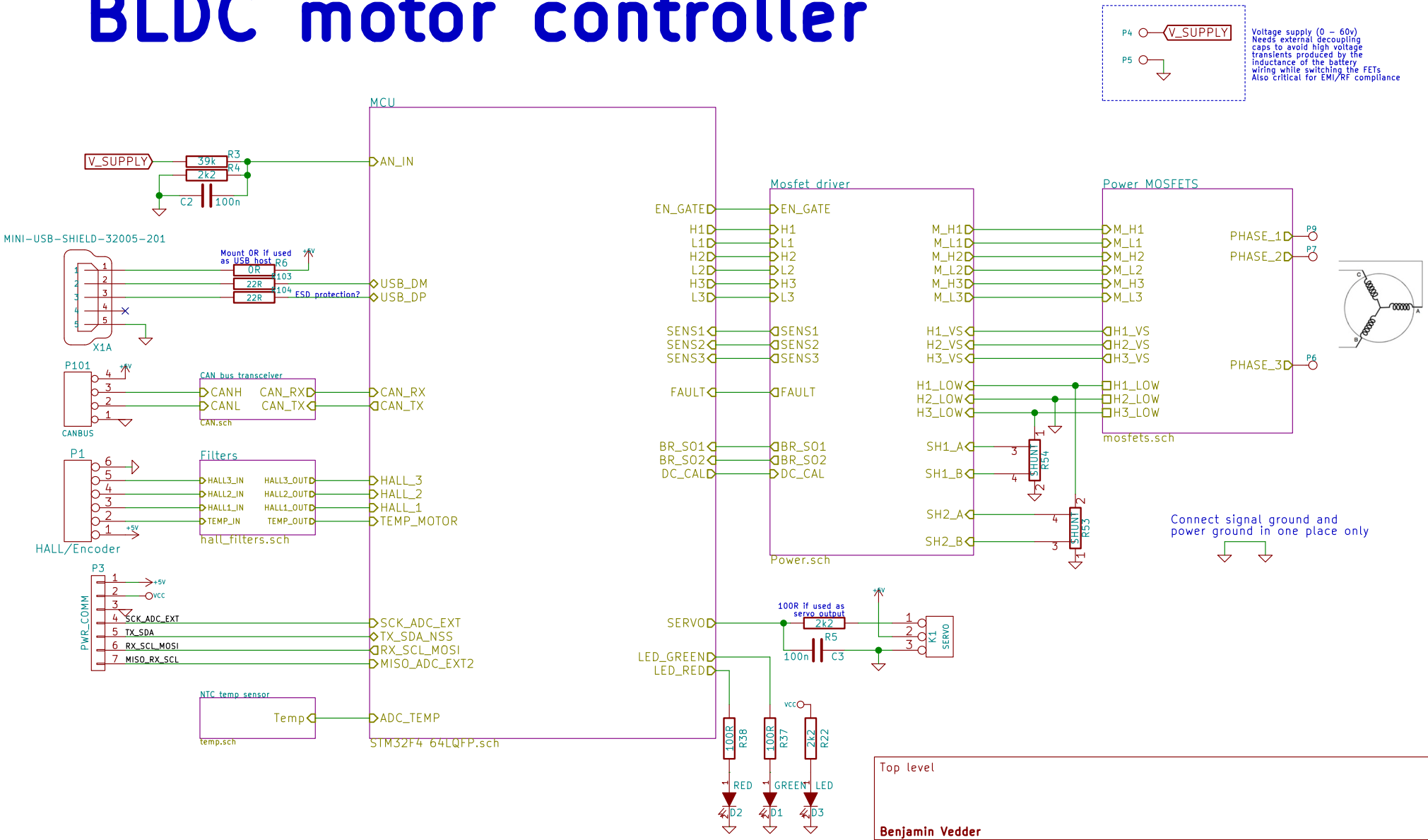
Throttle







BLDC motor controller



Drawings

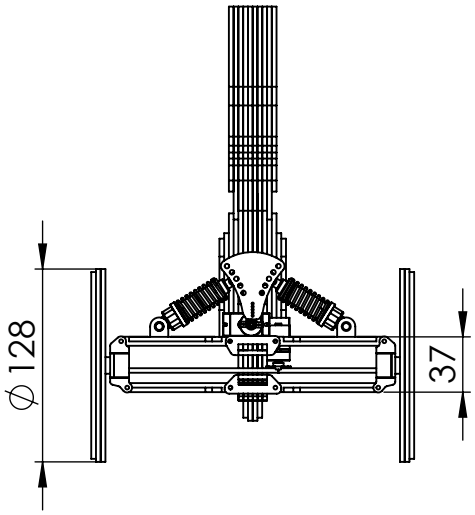
Small Scale Model - miniPEV

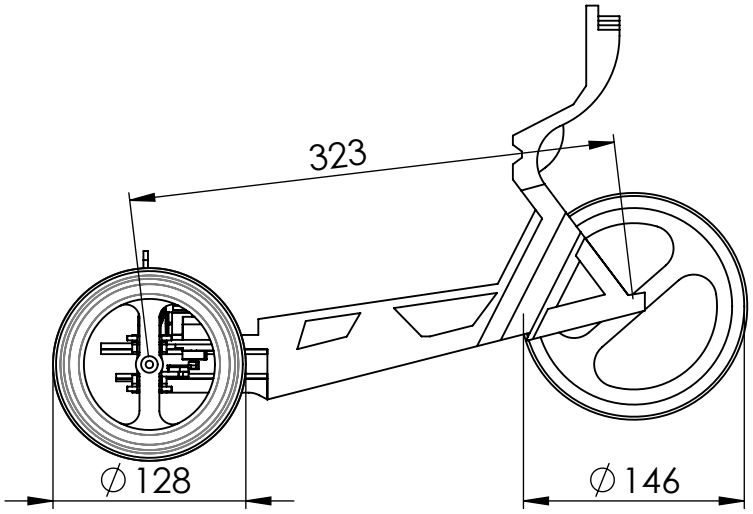
Real Scale Model - PEV

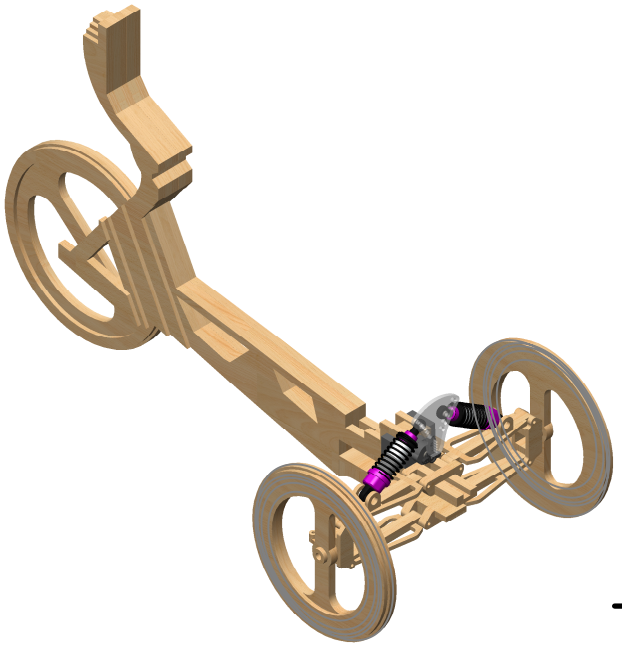
NIDEC 48R Motor

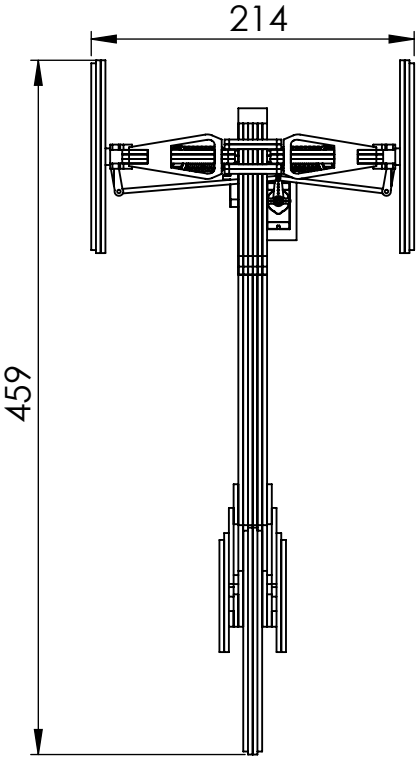
McMaster Components

236

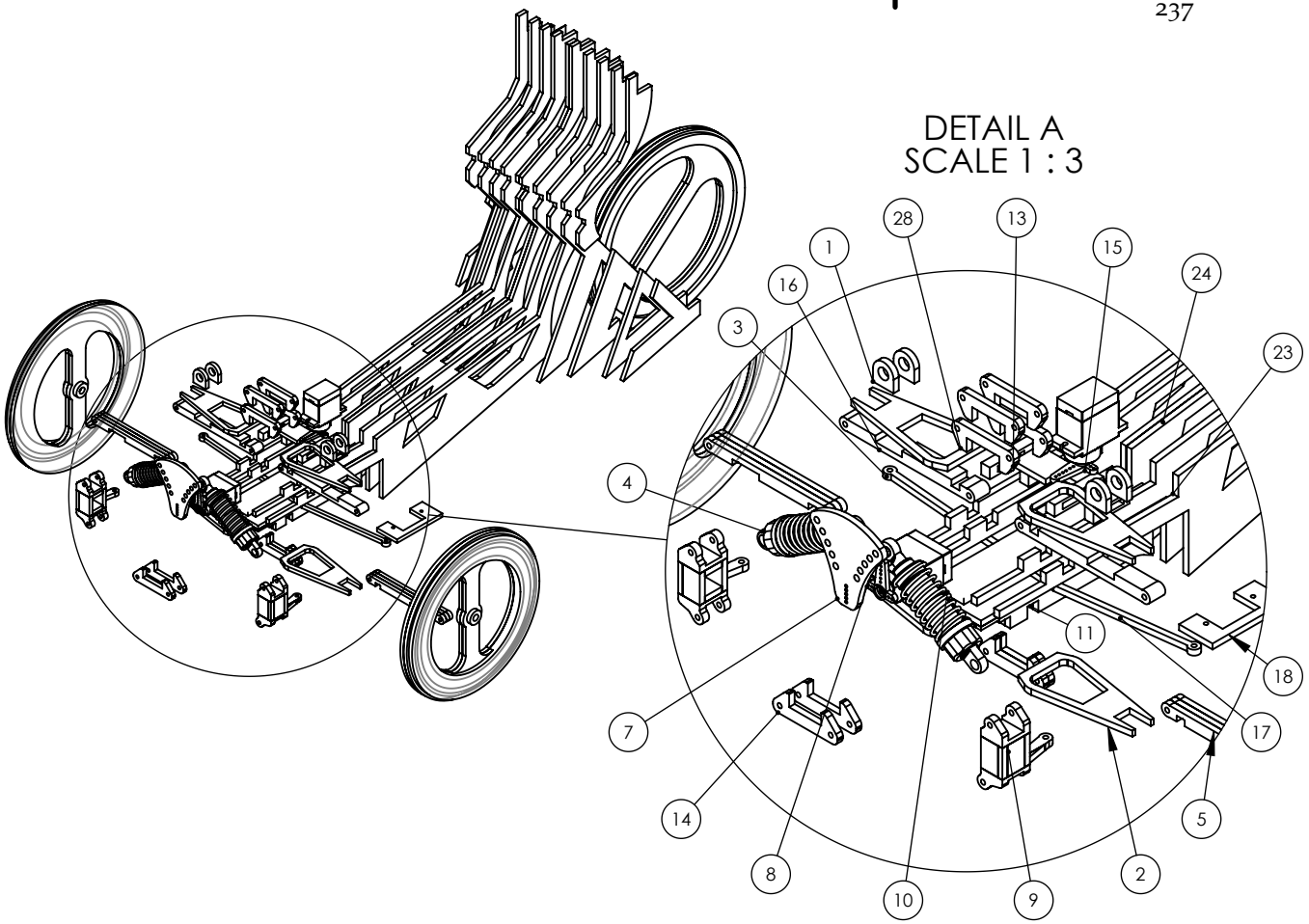








Material: **Material <not specified>**	This drawing and any information or descriptive material set out on it are the confidential and copyright property of MIT Media Lab. © and MUST NOT BE DISCLOSED, COPIED, LOANED in whole or part or used for any purpose without the written permission of Media Lab.	Description: **Assem1 - MiniPEV**	
Finish:			
Unless Otherwise Stated: Linear Tol.: ±**0.2**, Angular Tol.: **0°15'** Surface Finish: **0.8µm** All Dimensions: **mm**	Document Type: **Drawing**	Drawn by: **Iñigo Martinez**	Drawn Date: **26/04/2017**
Drawing Scale: **1:5**	**MIT Media Lab**	Checked/Approved by: **Michael Lin**	Checked/Approved Date: **26/04/2017**
Approx Weight: **160.08**gr / Drawing Produced In Accordance With: **BS8888**	Legal Owner: mit media lab 75 Amherst St, Cambridge, MA 02139	Part Number:	
Projection Method: **THIRD ANGLE** / Sheet Size: **A4**		Drawing Number: **1** / Sheet: **1 of 4** / Revision: **A**	



ITEM NO.	PART NUMBER	MATERIAL	WEIGHT	QTY.
1	Part16^Assem1	Balsa	0.05	4
2	Part11^Assem1	Balsa	0.33	4
3	Part20^Assem1	Material <not specified>	1.00	1
4	RC Damper HSP 296004	Material <not specified>	22.24	2
5	Part7^Assem1	Balsa	0.54	2
6	Part17^Assem1	Balsa	11.29	2
7	Part15^Assem1	Balsa	0.49	1
8	MG90servo_assembly_steering	Material <not specified>	0.03	1
9	Part13^Assem1	Balsa	1.38	1
10	Part1^Assem1	Balsa	1.06	1
11	Part5^Assem1	Balsa	0.38	6
12	Part4^Assem1	Balsa	0.97	1
13	Part9^Assem1	Balsa	0.14	1
14	Part6^Assem1	Balsa	0.14	4
15	MG90servo_assembly_steering_b	Material <not specified>	0.03	1
16	Part12^Assem1	Balsa	0.57	2
17	Part19^Assem1	Material <not specified>	0.70	1
18	Part18^Assem1	Material <not specified>	1.57	1
19	Part24^Assem1	Balsa	1.02	2
20	Part14^Assem1	Balsa	13.53	1
21	Part23^Assem1	Balsa	2.05	2
22	Part22^Assem1	Balsa	2.16	2
23	Part2^Assem1	Balsa	5.85	4
24	Part21^Assem1	Balsa	10.70	1
25	Part25^Assem1	Balsa	0.80	2
26	Part10^Assem1	Balsa	0.08	1
27	Part8^Assem1	Balsa	0.22	2
28	Part3^Assem1	Balsa	0.14	4

Material:
Material <not specified>

Finish:

Unless Otherwise Stated:
Linear Tol.: **±0.2**, Angular Tol.: **0°15'**
Surface Finish: **0.8µm**
All Dimensions: **mm**

Drawing Scale: **1:5**

Approx Weight: **160.08**gr

Drawing Produced In Accordance With: **BS8888**

Projection Method: **THIRD ANGLE**

Sheet Size: **A4**

Document Type: **Drawing**

MIT Media Lab

Legal Owner:

mit media lab

75 Amherst St, Cambridge, MA 02139

Description: **Assem1**

Drawn by: **Iñigo Martinez**

Drawn Date: **26/04/2017**

Checked/Approved by: **Michael Lin**

Checked/Approved Date: **26/04/2017**

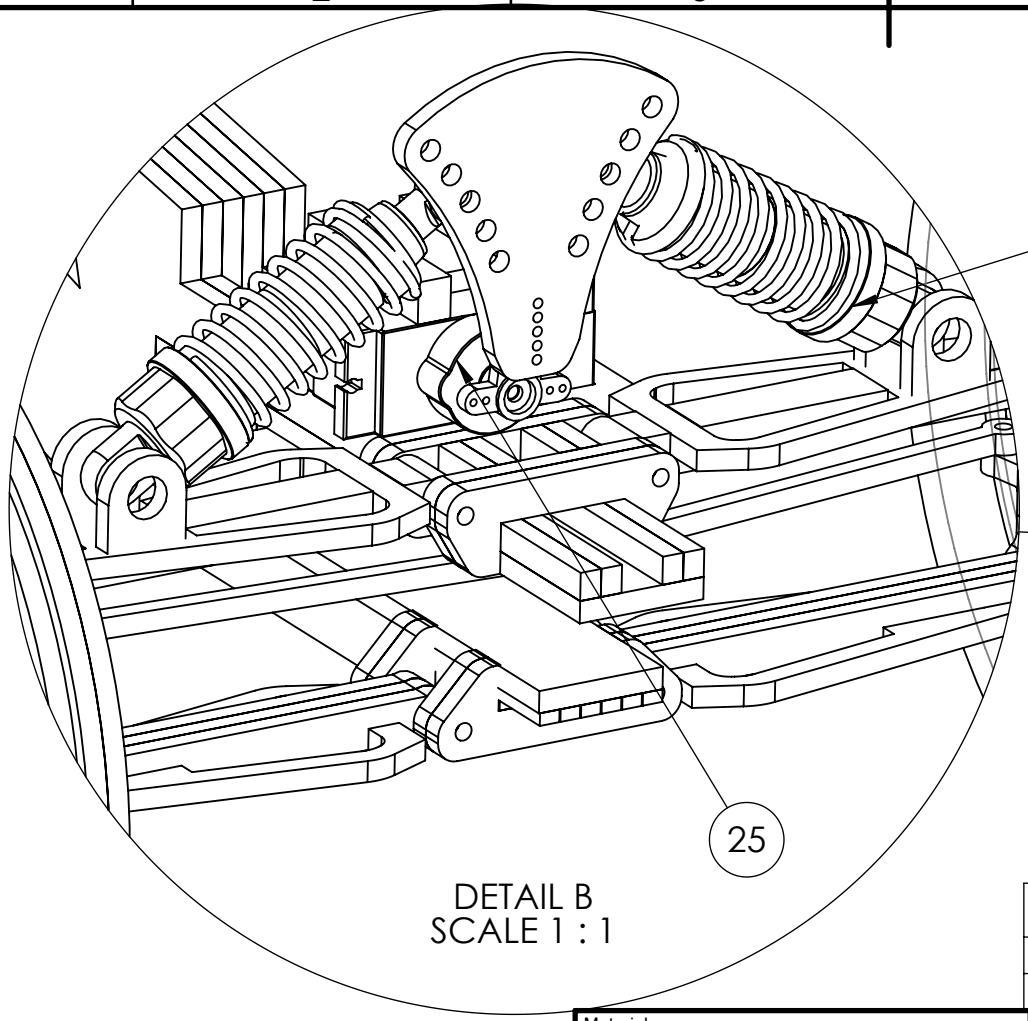
Part Number:

Drawing Number: **1**

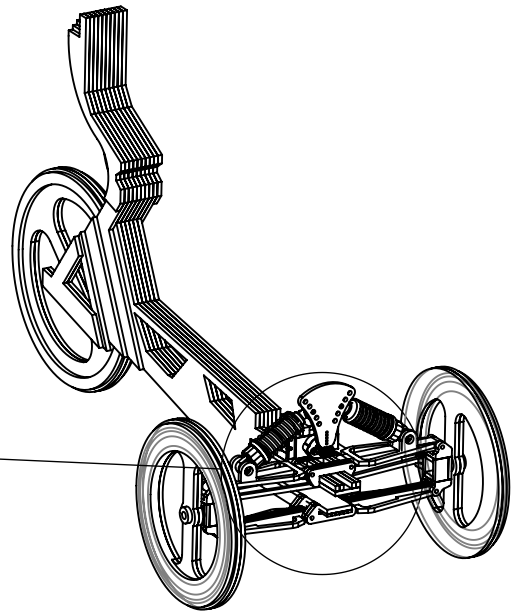
Sheet: **2 of 4**

Revision: **A**

238



DETAIL B
SCALE 1 : 1



ITEM NO.	PART NUMBER	DESCRIPTION	WEIGHT	QTY.
23	RC Damper HSP 296004	RC Damper HSP 296004	22.24	2
25	MG90servo_assembly_steering	MG90servo_assembly_steering	0.03	1

Material:
Material <not specified>

Finish:

Unless Otherwise Stated:
Linear Tol.: ±**0.2**, Angular Tol.: **0°15'**
Surface Finish: **0.8µm**
All Dimensions: **mm**

Drawing Scale: **1:5**

Approx Weight: **160.08**gr

Drawing Produced In Accordance With: **BS8888**

Projection Method: **THIRD ANGLE**

Sheet Size: **A4**

Document Type: **Drawing**

MIT Media Lab

Legal Owner:

mit media lab

75 Amherst St, Cambridge, MA 02139

Description: **Assem1 - MiniPEV**

Drawn by: **Iñigo Martinez**

Drawn Date: **26/04/2017**

Checked/Approved by: **Michael Lin**

Checked/Approved Date: **26/04/2017**

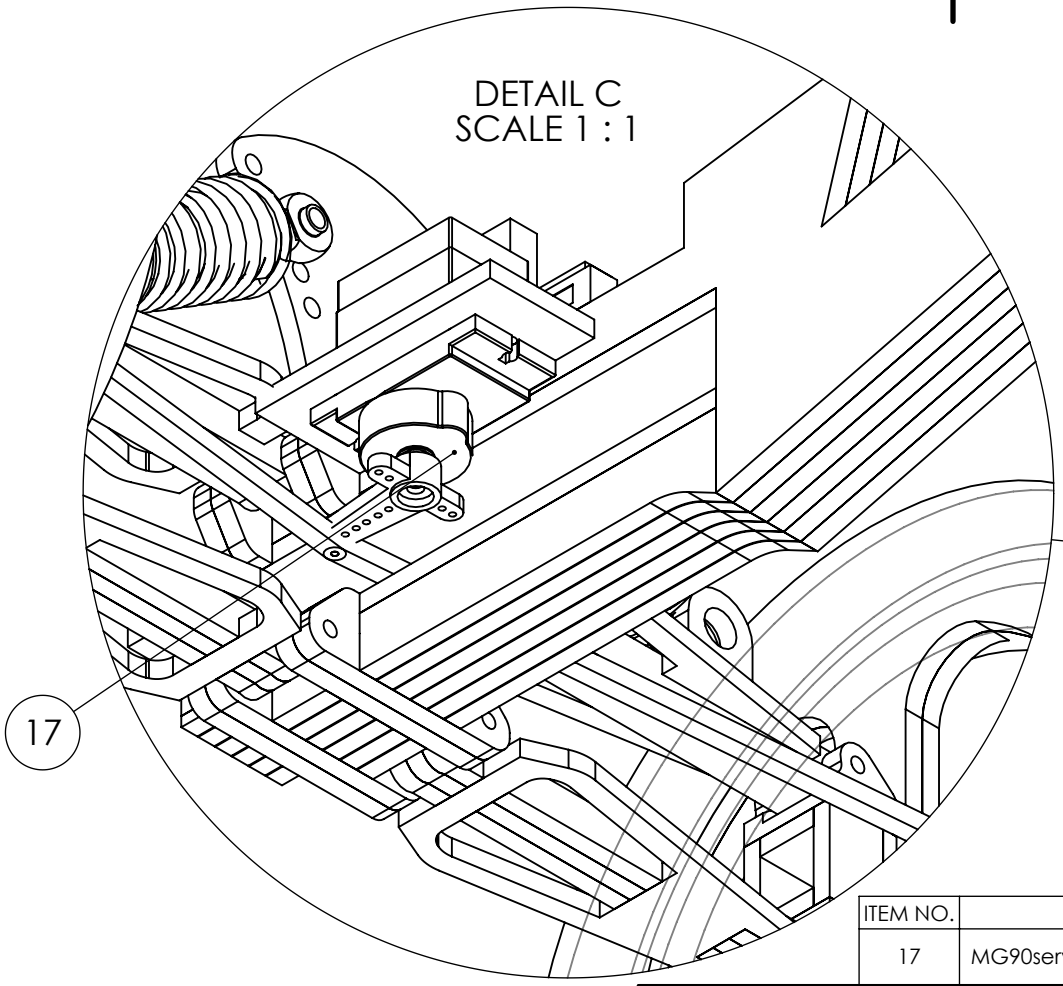
Part Number:

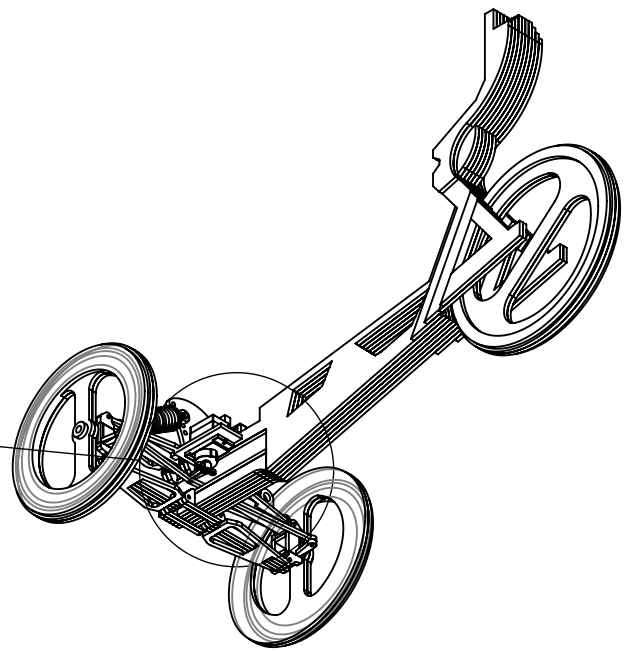
Drawing Number: **1**

Sheet: **3 of 4**

Revision: **A**

DETAIL C
SCALE 1 : 1





ITEM NO.	PART NUMBER	DESCRIPTION	MATERIAL	WEIGHT	QTY.
17	MG90servo_assembly_steering_b	MG90servo_assembly_steering_b	Material <not specified>	0.03	1

Material:
Material <not specified>

Finish:

Unless Otherwise Stated:
Linear Tol.: ±**0.2**, Angular Tol.: **0°15'**
Surface Finish: **0.8µm**
All Dimensions: **mm**

Drawing Scale: **1:5**

Approx Weight: **160.08**gr

Drawing Produced In Accordance With: **BS8888**

Projection Method: **THIRD ANGLE**

Sheet Size: **A4**

Document Type:
Drawing

MIT Media Lab

Legal Owner:
mit media lab
75 Amherst St, Cambridge, MA 02139

Description:
Assem1 - MiniPEV

Drawn by:
Iñigo Martinez

Drawn Date:
26/04/2017

Checked/Approved by:
Michael Lin

Checked/Approved Date:
26/04/2017

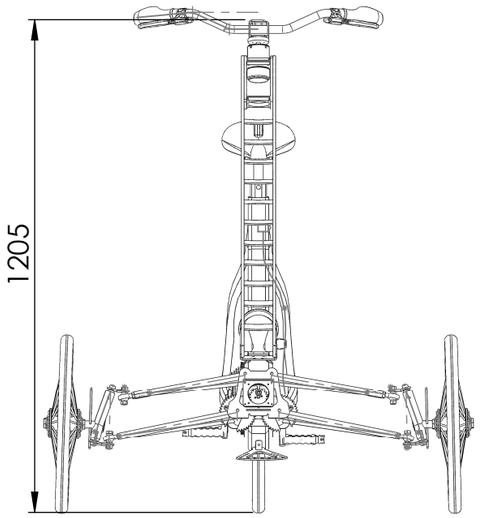
Part Number:

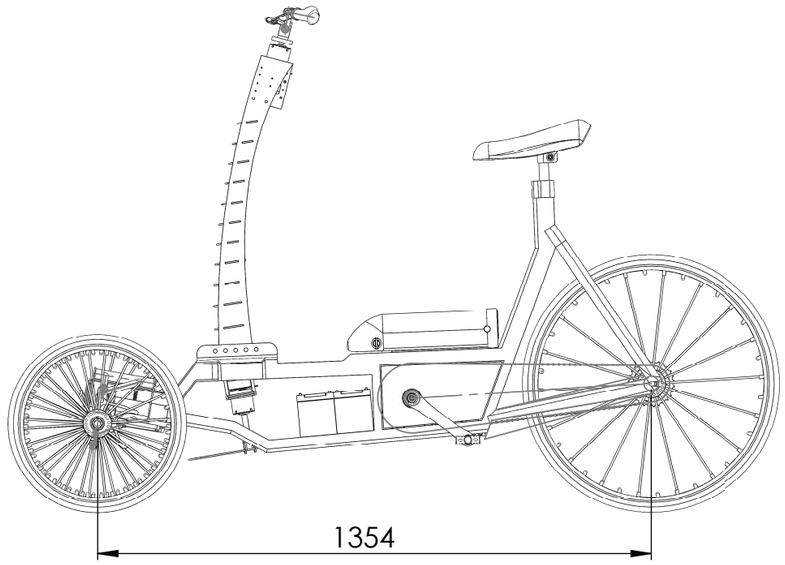
Drawing Number:
1

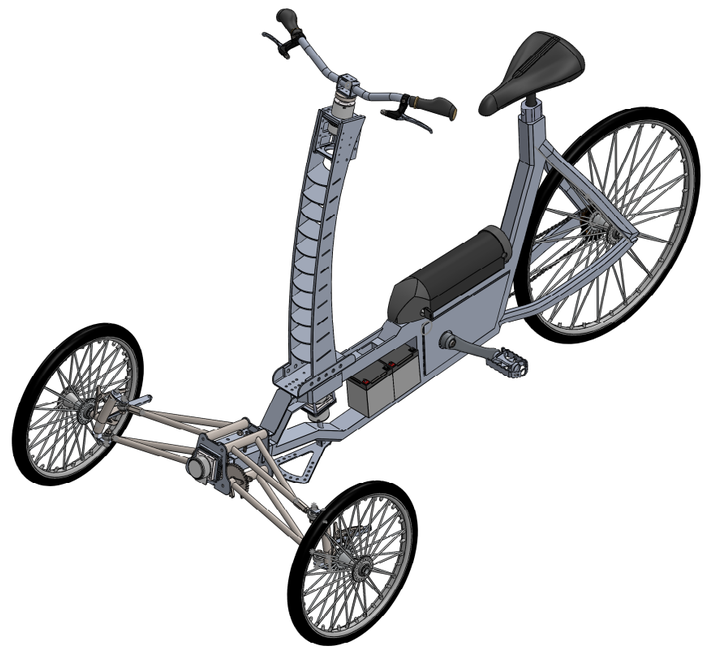
Sheet:
4 of 4

Revision:
A

240



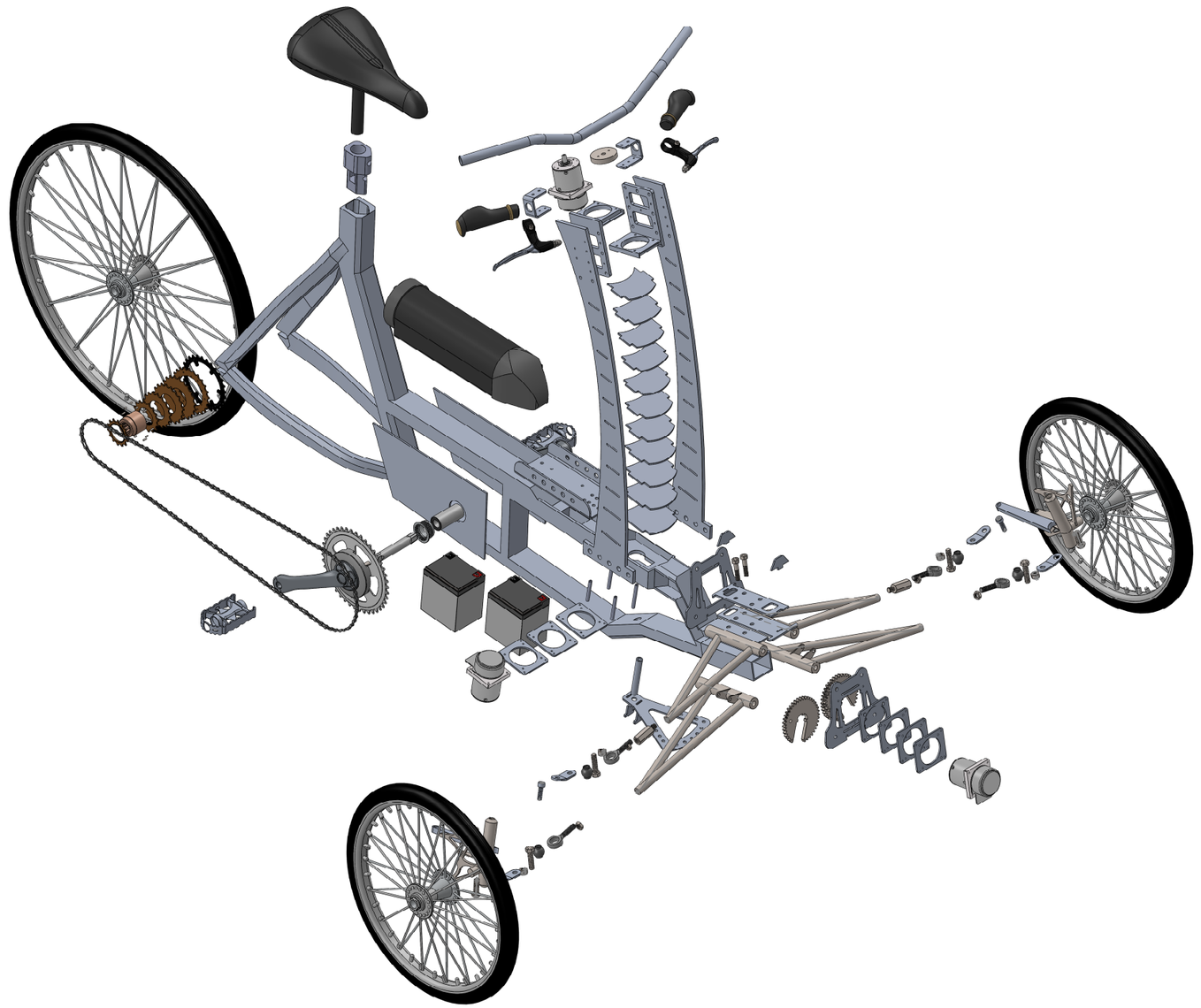




Material: **-**		This drawing and any information or descriptive material set out on it are the confidential and copyright property of **MIT Media Lab**. © and MUST NOT BE DISCLOSED, COPIED, LOANED in whole or part or used for any purpose without the written permission of **MIT Media Lab**.	Description: **Overall Design**
Finish:			
Unless Otherwise Stated: Linear Tol.: **±0.2**, Angular Tol.: **0°15'** Surface Finish: **0.8µm** All Dimensions: **mm**		Document Type: **Drawing**	Drawn by: **Iñigo Martinez** / Drawn Date: **27/04/2017**
Drawing Scale: **1:15**		**MIT Media Lab**	Checked/Approved by: **Michael Lin** / Checked/Approved Date: **27/04/2017**
Approx Weight: **37011.10** gr	Drawing Produced In Accordance With: **BS8888**	Legal Owner: mit media lab 75 Amherst St, Cambridge, MA 02139	Part Number:
Projection Method: **THIRD ANGLE**	Sheet Size: **A3**		Drawing Number: **1** / Sheet: **1 of 6** / Revision: **A**

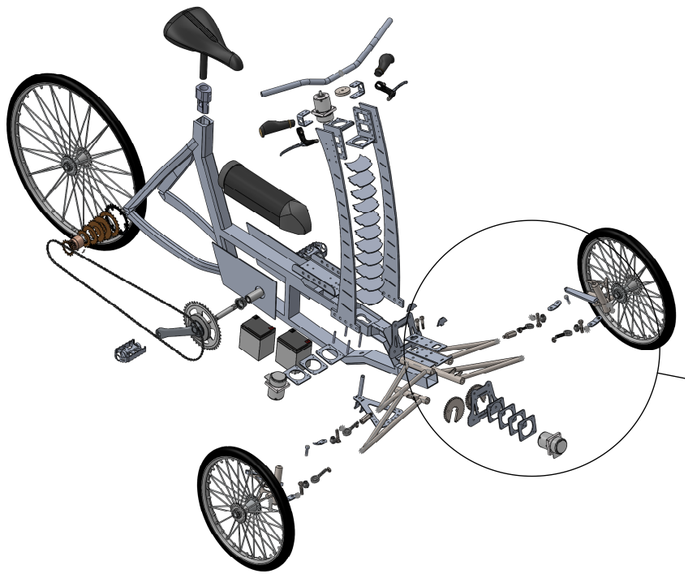
ITEM NO.	DESCRIPTION	MATERIAL	WEIGHT	Default/ QTY.
1	Frame^Redesign_motor_02_01	1060 Alloy	12064.65	1
2	Bracket_R^Redesign_motor_02_01	1060 Alloy	79.08	1
3	Bracket_F^Redesign_motor_02_01	1060 Alloy	83.38	1
4	Bridge_Low_1^Redesign_motor_02_01	1060 Alloy	48.60	1
5	Bridge_Low_2^Redesign_motor_02_01	1060 Alloy	56.60	1
6	Join_Low_1^Redesign_motor_02_01	1060 Alloy	5.61	1
7	Join_Low_2^Redesign_motor_02_01	1060 Alloy	5.31	1
8	90201A228_EXTREME-STRENGTH GRADE 9 STEEL CAP SCREW	Alloy Steel	0.04	2
9	TL_Arm^Redesign_motor_02_01	Alloy Steel	1003.01	1
10	6960T610_SUPER-SWIVEL BALL JOINT ROD END - fix	Alloy Steel	0.093	4
11	6960T610_SUPER-SWIVEL BALL JOINT ROD END - ball	Alloy Steel	0.059	4
12	94846A515_GRADE 5 STEEL THIN HEX NUT	Alloy Steel	0.01	4
13	92499A298_18-8 SS MALE-FEM HEX THRD ADAPTER	Alloy Steel	0.13	2
14	90499A815_GRADE 8 STEEL HEX NUT	Alloy Steel	0.016	4
15	92620A628_HIGH-STRENGTH GRADE 8 STEEL CAP SCREW	Alloy Steel	0.06	4
16	BL_Arm^Redesign_motor_02_01	Alloy Steel	1153.57	1
17	BR_Arm^Redesign_motor_02_01	Alloy Steel	1153.60	1
18	TR_Arm^Redesign_motor_02_01	Alloy Steel	1003.01	1
19	Kingpin_L^Redesign_motor_02_01	Alloy Steel	513.78	1
20	Align_top^Redesign_motor_02_01	1060 Alloy	8.52	1
21	Align_bottom^Redesign_motor_02_01	1060 Alloy	9.22	1
22	91239A515_CLASS 10.9 STL BUTTON-HEAD SOCKET CAP SCREW	Alloy Steel	21.578	2
23	91292A148_TYPE 18-8 SS SOCKET HEAD CAP SCREW	1060 Alloy	5.19	2
24	Kingpin_L_Steering^Redesign_motor_02_01	1060 Alloy	63.80	1
25	Kingpin_R^Redesign_motor_02_01	Alloy Steel	513.93	1
26	Copy of Align_top^Redesign_motor_02_01	1060 Alloy	8.52	1
27	Copy of Align_bottom^Redesign_motor_02_01	1060 Alloy	9.22	1
28	Kingpin_R_Steering^Redesign_motor_02_01	1060 Alloy	71.68	1
29	TIRE WHEEL ASSEM	Material <not specified>	6.61	2
30	TIRE WHEEL ASSEM_Rear	Material <not specified>	8.82	1
31	80-A096-0004	Material <not specified>	104.21	3
32	Copy of Join_motor	1060 Alloy	18.39	1
33	Join_motor^Redesign_motor_02_01	1060 Alloy	22.82	6
34	Gear_Blank_new	Alloy Steel	65.75	1
35	Gear_Blank2_new	Alloy Steel	207.83	2
36	Steering_tube^Redesign_motor_02_01	1060 Alloy	14.35	1
37	Steering_Join	Material <not specified>	72.76	1
38	Steering_Motor_Tubes^Redesign_motor_02_01	Material <not specified>	4.44	1
39	60645K131_STEEL BALL JOINT ROD END_Ball	Alloy Steel	0.022	4
40	Bike_Seat	Material <not specified>	87.72	1
41	SEAT	Material <not specified>	1.00	1
42	kett_115-590-50	Plain Carbon Steel	200.00	1
43	Crank_assembledpartst	1060 Alloy	400.00	1
44	Crank_assembledpartst2	1060 Alloy	200.00	1
45	SpRocket	Material <not specified>	0.44	1
46	Shimano UN52BB	Material <not specified>	120.00	1
47	Pedal	1060 Alloy	165.00	2
48	Drivetrain_Cover_R^Redesign_motor_02_01	1060 Alloy	311.94	1
49	Drivetrain_Cover_L^Redesign_motor_02_01	1060 Alloy	314.32	1
50	Steering_Bar_Support^Redesign_motor_02_01	1060 Alloy	241.22	1
51	Steering_Bar^Redesign_motor_02_01	1060 Alloy	776.93	2
52	Steering_Bar_Join_1^Redesign_motor_02_01	1060 Alloy	53.89	1
53	Steering_Bar_Join_2^Redesign_motor_02_01	1060 Alloy	50.75	1
54	Steering_Bar_Join_3^Redesign_motor_02_01	1060 Alloy	47.01	1
55	Steering_Bar_Join_4^Redesign_motor_02_01	1060 Alloy	44.48	1
56	Steering_Bar_Join_5^Redesign_motor_02_01	1060 Alloy	40.83	1
57	Steering_Bar_Join_6^Redesign_motor_02_01	1060 Alloy	37.91	1
58	Steering_Bar_Join_7^Redesign_motor_02_01	1060 Alloy	36.54	1
59	Steering_Bar_Join_8^Redesign_motor_02_01	1060 Alloy	33.68	1
60	Steering_Bar_Join_9^Redesign_motor_02_01	1060 Alloy	32.87	1
61	Steering_Bar_Join_10^Redesign_motor_02_01	1060 Alloy	37.45	1
62	Steering_Bar_Join_11^Redesign_motor_02_01	1060 Alloy	35.74	1
63	Handle_Motor_Join^Redesign_motor_02_01	1060 Alloy	148.80	2
64	Handle_Bar^Redesign_motor_02_01	1060 Alloy	141.55	1
65	Handle_Grip_Brake^Redesign_motor_02_01	Material <not specified>	181.48	1
66	Handle_Motor_Gear^Redesign_motor_02_01	Alloy Steel	132.71	1
67	Handle_Bar_Attachment_1^Redesign_motor_02_01	1060 Alloy	23.80	1
68	Handle_Bar_Attachment_2^Redesign_motor_02_01	1060 Alloy	22.68	1
69	Battery12V_5A	Material <not specified>	633.76	2
70	accupack	Material <not specified>	565.60	1

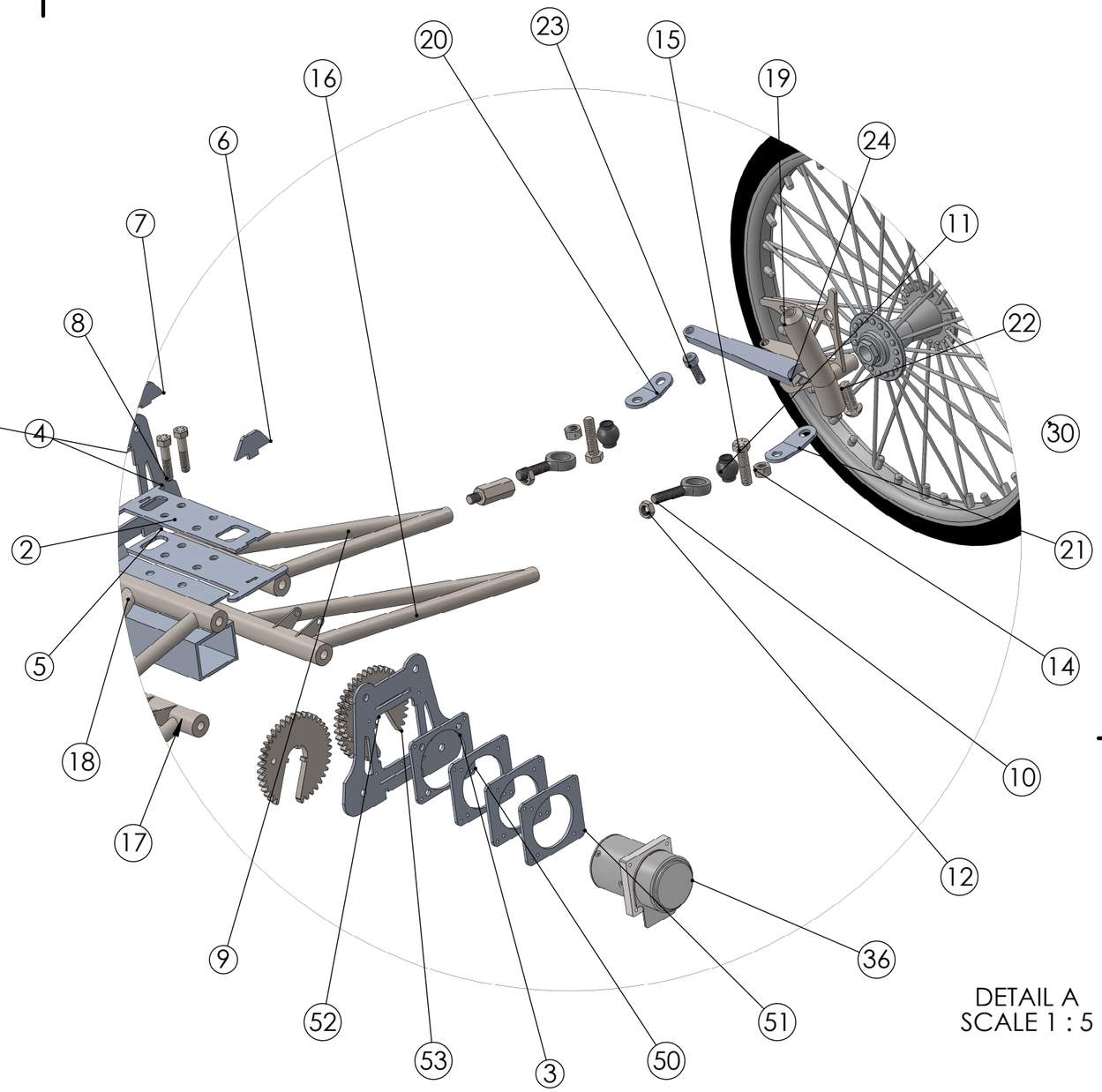
241



Material: -	This drawing and any information or descriptive material set out on it are the confidential and copyright property of **MIT Media Lab**. © and MUST NOT BE DISCLOSED, COPIED, LOANED in whole or part or used for any purpose without the written permission of **MIT Media Lab**.	Description: **Exploded Model + Bill of Materials**	
Finish:		Drawn by: **Iñigo Martinez**	Drawn Date: **27/04/2017**
Unless Otherwise Stated: Linear Tol.: **±0.2**, Angular Tol.: **0°15'** Surface Finish: **0.8µm** All Dimensions: **mm**	Document Type: **Drawing**	Checked/Approved by: **Michael Lin**	Checked/Approved Date: **27/04/2017**
Drawing Scale: **1:10**	**MIT Media Lab**		
Approx Weight: **37011.10**gr / Drawing Produced In Accordance With: **BS8888**	Legal Owner: mit media lab 75 Amherst St, Cambridge, MA 02139	Part Number:	
Projection Method: **THIRD ANGLE** / Sheet Size: **A3**		Drawing Number: **1**	Sheet: **2 of 6** / Revision: **A**

242





DETAIL A
SCALE 1 : 5

ITEM NO.	DESCRIPTION	MATERIAL	WEIGHT	QTY.
2	Bracket_R^Redesign_motor_02_01	1060 Alloy	79.08	1
3	Bracket_F^Redesign_motor_02_01	1060 Alloy	83.38	1
4	Bridge_Low_1^Redesign_motor_02_01	1060 Alloy	48.60	1
5	Bridge_Low_2^Redesign_motor_02_01	1060 Alloy	56.60	1
6	Join_Low_1^Redesign_motor_02_01	1060 Alloy	5.61	1
7	Join_Low_2^Redesign_motor_02_01	1060 Alloy	5.31	1
8	90201A228_EXTREME-STRENGTH GRADE 9 STEEL CAP SCREW	Alloy Steel	0.04	2
10	6960T610_SUPER-SWIVEL BALL JOINT ROD END - fix	Alloy Steel	0.093	4
11	6960T610_SUPER-SWIVEL BALL JOINT ROD END - ball	Alloy Steel	0.059	4
12	94846A515_GRADE 5 STEEL THIN HEX NUT	Alloy Steel	0.01	4
14	90499A815_GRADE 8 STEEL HEX NUT	Alloy Steel	0.016	4
15	92620A628_HIGH-STRENGTH GRADE 8 STEEL CAP SCREW	Alloy Steel	0.06	4
16	BL_Arm^Redesign_motor_02_01	Alloy Steel	1153.57	1
17	BR_Arm^Redesign_motor_02_01	Alloy Steel	1153.60	1
18	TR_Arm^Redesign_motor_02_01	Alloy Steel	1003.01	1
19	Kingpin_L^Redesign_motor_02_01	Alloy Steel	513.78	1
20	Align_top^Redesign_motor_02_01	1060 Alloy	8.52	1
21	Align_bottom^Redesign_motor_02_01	1060 Alloy	9.22	1
22	91239A515_CLASS 10.9 STL BUTTON-HEAD SOCKET CAP SCREW	Alloy Steel	21.578	2
23	91292A148_TYPE 18-8 SS SOCKET HEAD CAP SCREW	1060 Alloy	5.19	2
24	Kingpin_L_Steering^Redesign_motor_02_01	1060 Alloy	63.80	1
30	Tire	Natural Rubber	534.21	2
36	_X2_96F64EF6_X0_4	Material <not specified>	12.84	3
50	Copy of Join_motor	1060 Alloy	18.39	1
51	Join_motor^Redesign_motor_02_01	1060 Alloy	22.82	6
52	Gear_Blank_new	Alloy Steel	65.75	1
53	Gear_Blank2_new	Alloy Steel	207.83	2

Material: -

Finish:

Unless Otherwise Stated:
Linear Tol.: ±**0.2**, Angular Tol.: **0°15'**
Surface Finish: **0.8µm**
All Dimensions: **mm**

Drawing Scale: **1:20**

Approx Weight: **37011.10** gr

Drawing Produced In Accordance With: **BS8888**

Projection Method: **THIRD ANGLE**

Sheet Size: **A3**

Document Type: **Drawing**

MIT Media Lab

Legal Owner:

mit media lab

75 Amherst St, Cambridge, MA 02139

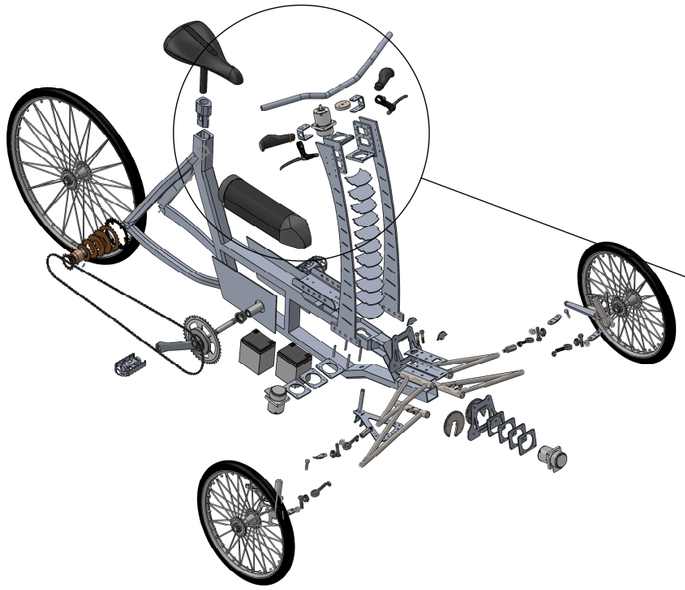
Description: **Suspension + Tilting Detailed View**

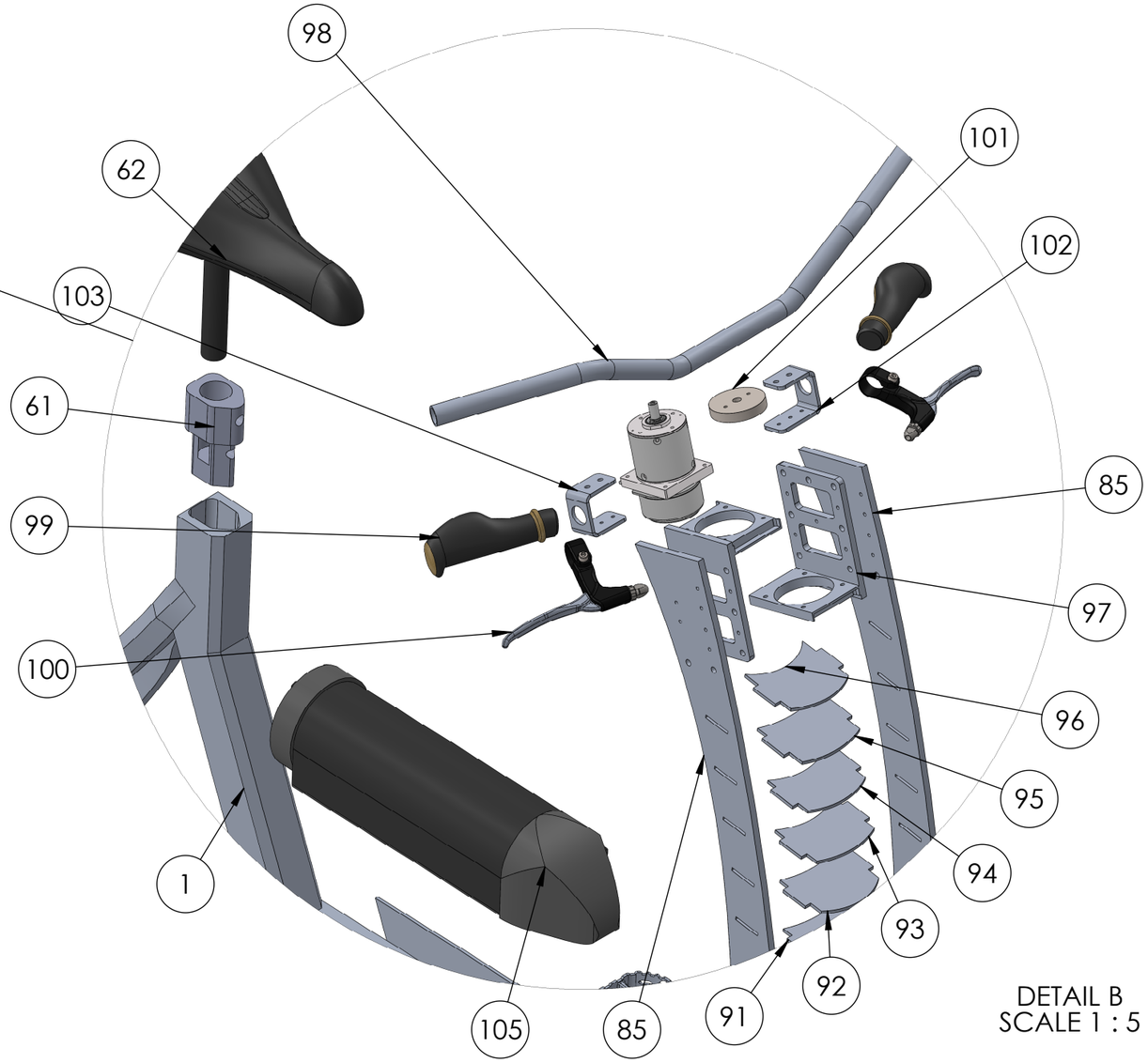
Drawn by: **Iñigo Martinez** | Drawn Date: **27/04/2017**

Checked/Approved by: **Michael Lin** | Checked/Approved Date: **27/04/2017**

Part Number:

Drawing Number: **1** | Sheet: **3 of 6** | Revision: **A**





DETAIL B
SCALE 1 : 5

ITEM NO.	DESCRIPTION	MATERIAL	WEIGHT	QTY.
1	Frame^Redesign_motor_02_01	1060 Alloy	12064.65	1
61	Part2^Bike_Seat	Material <not specified>	68.70	1
62	SEAT	Material <not specified>	1.00	1
85	Steering_Bar^Redesign_motor_02_01	1060 Alloy	776.93	2
91	Steering_Bar_Join_6^Redesign_motor_02_01	1060 Alloy	37.91	1
92	Steering_Bar_Join_7^Redesign_motor_02_01	1060 Alloy	36.54	1
93	Steering_Bar_Join_8^Redesign_motor_02_01	1060 Alloy	33.68	1
94	Steering_Bar_Join_9^Redesign_motor_02_01	1060 Alloy	32.87	1
95	Steering_Bar_Join_10^Redesign_motor_02_01	1060 Alloy	37.45	1
96	Steering_Bar_Join_11^Redesign_motor_02_01	1060 Alloy	35.74	1
97	Handle_Motor_Join^Redesign_motor_02_01	1060 Alloy	148.80	2
98	Handle_Bar^Redesign_motor_02_01	1060 Alloy	141.55	1
99	stang	Material <not specified>	136.07	1
100	stang2	Material <not specified>	45.41	1
101	Handle_Motor_Gear^Redesign_motor_02_01	Alloy Steel	132.71	1
102	Handle_Bar_Attachment_1^Redesign_motor_02_01	1060 Alloy	23.80	1
103	Handle_Bar_Attachment_2^Redesign_motor_02_01	1060 Alloy	22.68	1
105	accupack	Material <not specified>	565.60	1

Material: -

Finish:

Unless Otherwise Stated:
Linear Tol.: ±**0.2**, Angular Tol.: **0°15'**
Surface Finish: **0.8µm**
All Dimensions: **mm**

Drawing Scale: **1:20**

Approx Weight: **37011.10** gr

Drawing Produced In Accordance With: **BS8888**

Projection Method: **THIRD ANGLE**

Sheet Size: **A3**

Document Type: **Drawing**

MIT Media Lab

Legal Owner:

mit media lab
75 Amherst St, Cambridge, MA 02139

Description: **Handle-Bar Detailed View**

Drawn by: **Iñigo Martinez**

Drawn Date: **27/04/2017**

Checked/Approved by: **Michael Lin**

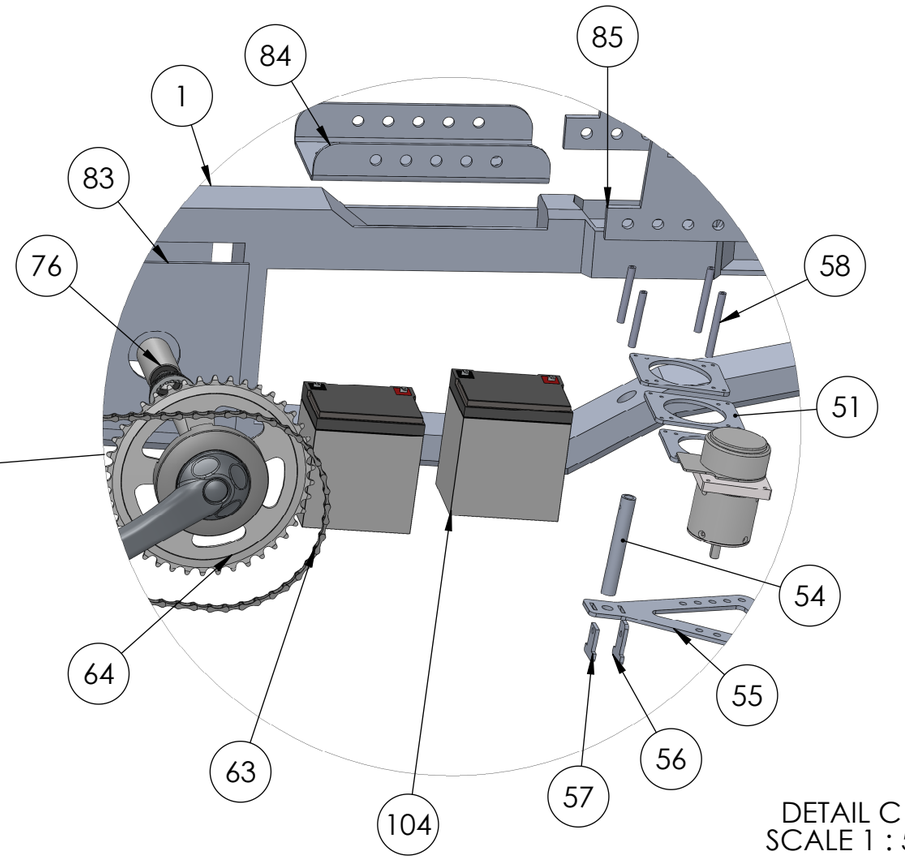
Checked/Approved Date: **27/04/2017**

Part Number:

Drawing Number: **1**

Sheet: **4 of 6**

Revision: **A**



DETAIL C
SCALE 1 : 5

ITEM NO.	DESCRIPTION	MATERIAL	WEIGHT	QTY.
1	Frame^Redesign_motor_02_01	1060 Alloy	12064.65	1
51	Join_motor^Redesign_motor_02_01	1060 Alloy	22.82	6
54	Steering_tube^Redesign_motor_02_01	1060 Alloy	14.35	1
55	Part1	1060 Alloy	64.36	1
56	Part2^Steering_Join	1060 Alloy	4.57	1
57	Copy of Part2^Steering_Join	1060 Alloy	3.83	1
58	Steering_Motor_Tubes^Redesign_motor_02_01	Material <not specified>	4.44	1
63	kett_115-590-50	Plain Carbon Steel	200.00	1
64	Crank_assembledpartst	1060 Alloy	400.00	1
76	UN52-cartridge	Chrome Stainless Steel	190.43	1
83	Drivetrain_Cover_L^Redesign_motor_02_01	1060 Alloy	314.32	1
84	Steering_Bar_Support^Redesign_motor_02_01	1060 Alloy	241.22	1
85	Steering_Bar^Redesign_motor_02_01	1060 Alloy	776.93	2
104	Battery12V_5A	Material <not specified>	633.76	2

Material: -

Finish:

Unless Otherwise Stated:
Linear Tol.: ±**0.2**, Angular Tol.: **0°15'**
Surface Finish: **0.8µm**
All Dimensions: **mm**

Drawing Scale: **1:20**

Approx Weight: **37011.10** gr

Drawing Produced In Accordance With: **BS8888**

Projection Method: **THIRD ANGLE**

Sheet Size: **A3**

Document Type: **Drawing**

MIT Media Lab

Legal Owner:

mit media lab
75 Amherst St, Cambridge, MA 02139

Description: **Steering + Batteries Detailed View**

Drawn by: **Iñigo Martinez**

Drawn Date: **27/04/2017**

Checked/Approved by: **Michael Lin**

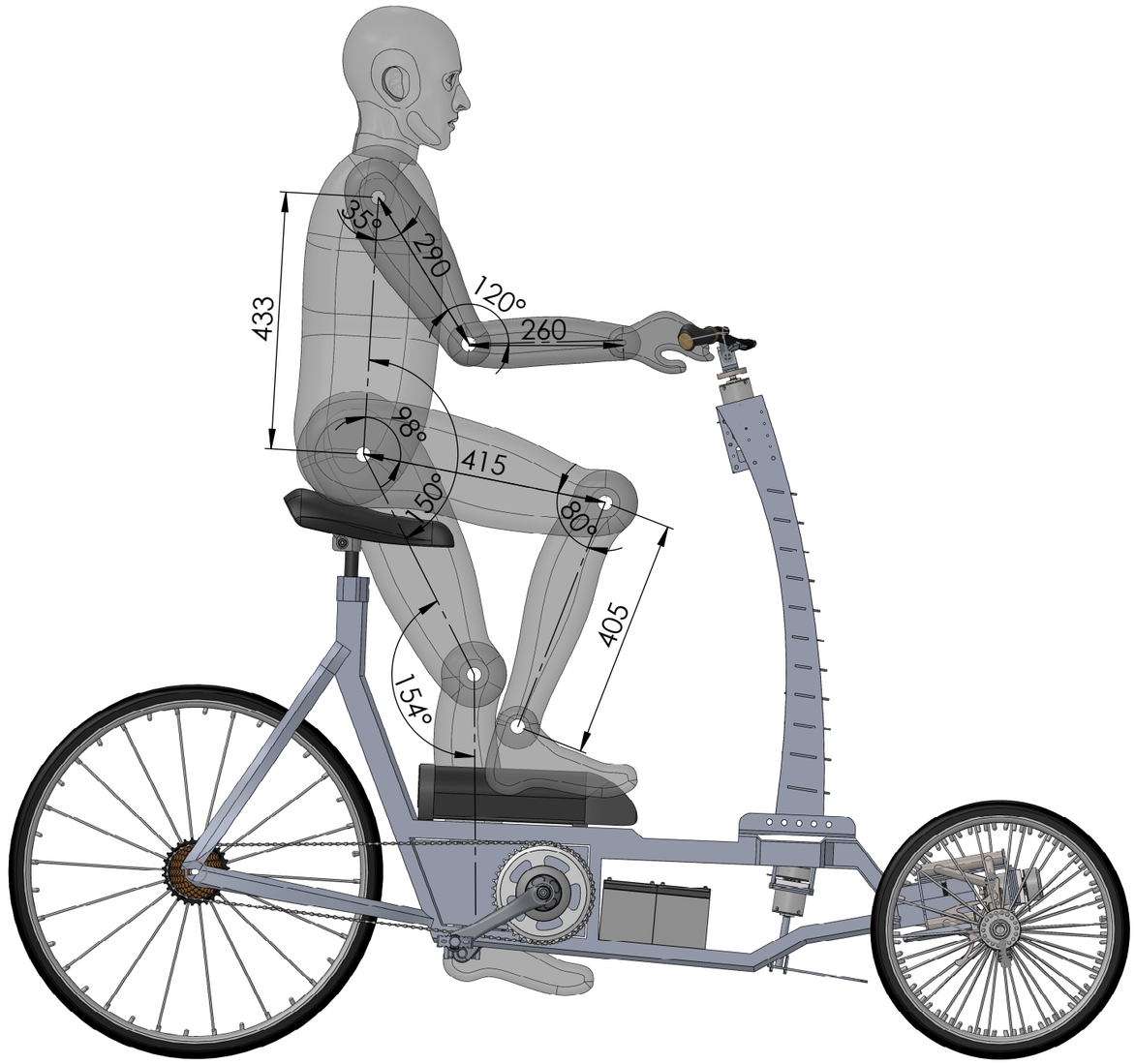
Checked/Approved Date: **27/04/2017**

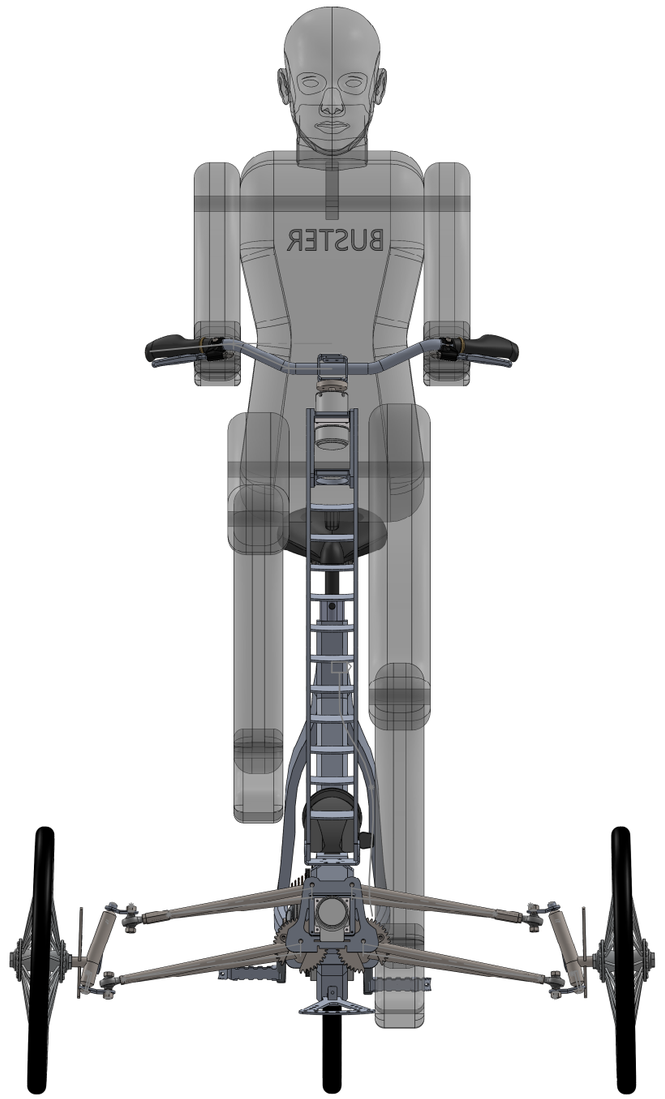
Part Number:

Drawing Number: **1**

Sheet: **5 of 6**

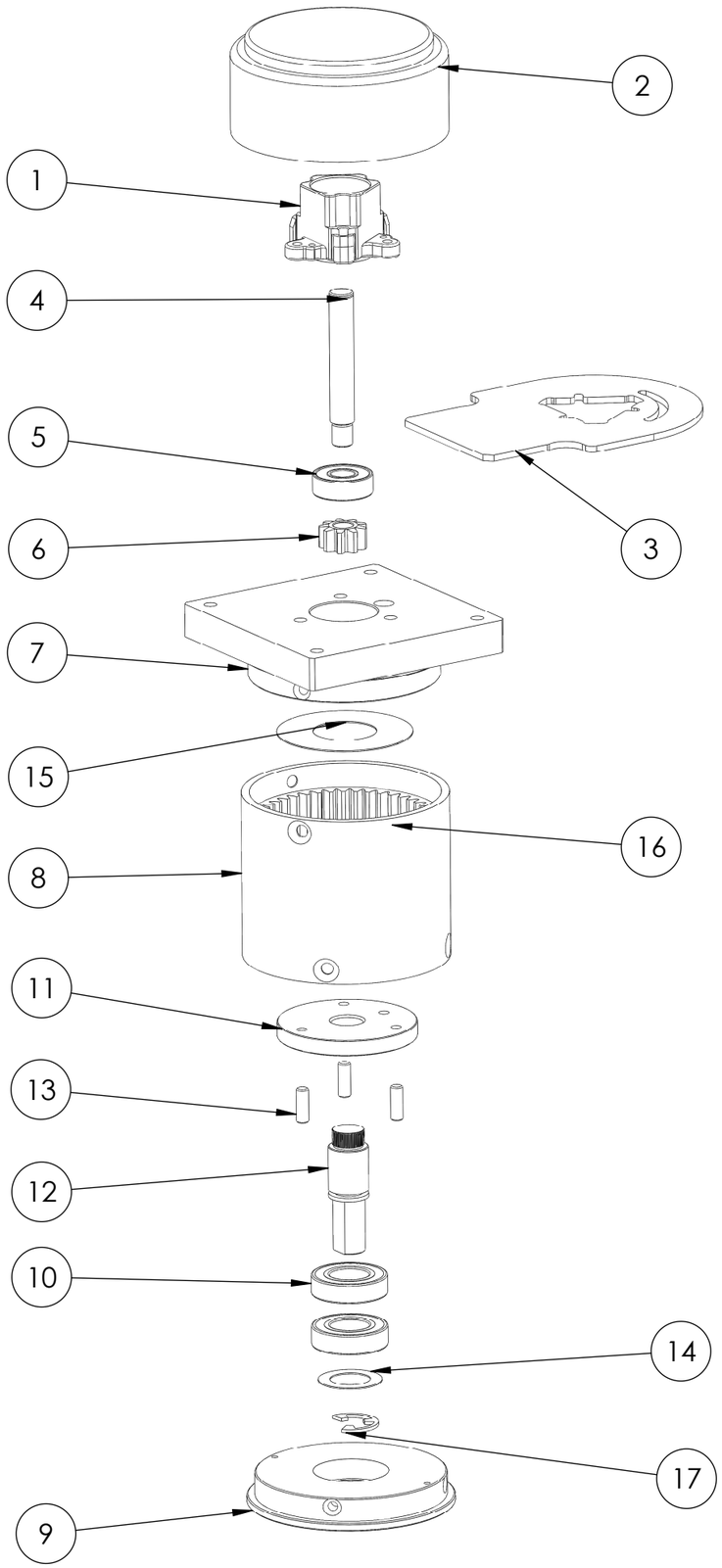
Revision: **A**

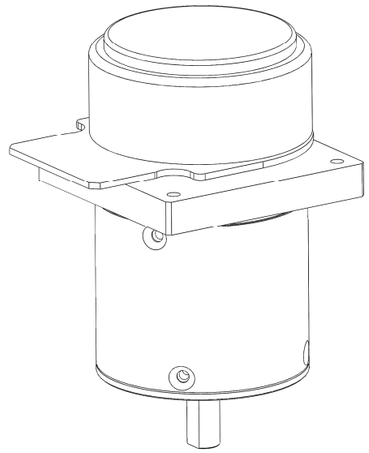




Material: **-**	This drawing and any information or descriptive material set out on it are the confidential and copyright property of **MIT Media Lab**. © and MUST NOT BE DISCLOSED, COPIED, LOANED in whole or part or used for any purpose without the written permission of **MIT Media Lab**.	Description: **Ergonomics Design with Driver Model**	
Finish:		Drawn by: **Iñigo Martinez**	Drawn Date: **27/04/2017**
Unless Otherwise Stated: Linear Tol.: ±**0.2**, Angular Tol.: **0°15'** Surface Finish: **0.8µm** All Dimensions: **mm**	Document Type: **Drawing**	Checked/Approved by: **Michael Lin**	Checked/Approved Date: **27/04/2017**
Drawing Scale: **1:10**	**MIT Media Lab**		
Approx Weight: **37011.10g** / Drawing Produced In Accordance With: **BS8888**	Legal Owner: mit media lab 75 Amherst St, Cambridge, MA 02139	Part Number:	
Projection Method: **THIRD ANGLE** / Sheet Size: **A3**		Drawing Number: **1**	Sheet: **6 of 6** / Revision: **A**

246





ITEM NO.	DESCRIPTION	WEIGHT	QTY.
1	_X2_96F64EF6_X0_2	2.72	1
2	_X2_96F64EF6_X0_4	12.84	1
3	_X2_96F64EF6_X0_8	4.43	1
4	_X2_96F64EF6_X0_10	1.07	1
5	_X2_96F64EF6_X0_12	0.29	1
6	9G-A036-0003	0.35	1
7	9T-A096-0006	38.57	1
8	9G-A036-0018	22.76	1
9	9T-A036-0029	13.09	1
10	9S-104-0004	1.01	2
11	9T-A036-0014	3.80	1
12	9T-A089-0013	1.80	1
13	9S-A036-0002	0.06	3
14	9P-A036-0004	0.04	1
15	9P-A036-0003	0.21	1
16	9P-A036-0001	0.00	4
17	9S-103-0002	0.04	1

Material: -

Finish:

Unless Otherwise Stated:
Linear Tol.: **±0.2**, Angular Tol.: **0°15'**
Surface Finish: **0.8µm**
All Dimensions: **mm**

Drawing Scale: **2:3**

Approx Weight: **104.21**gr

Drawing Produced In Accordance With: **BS8888**

Projection Method: **THIRD ANGLE**

Sheet Size: **A3**

Document Type: **Drawing**

MIT Media Lab

Legal Owner:

mit media lab
75 Amherst St, Cambridge, MA 02139

Description: **NIDEC 48R Motor T-IT-N**

Drawn by: **Iñigo Martinez**

Drawn Date: **27/04/2017**

Checked/Approved by: **Michael Lin**

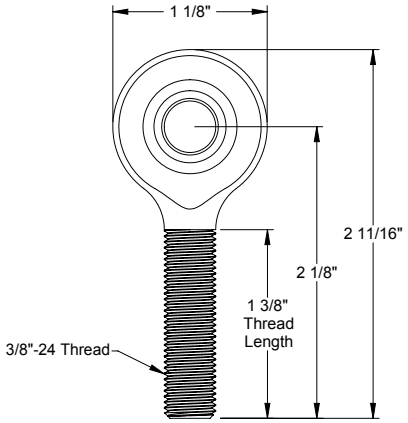
Checked/Approved Date: **27/04/2017**

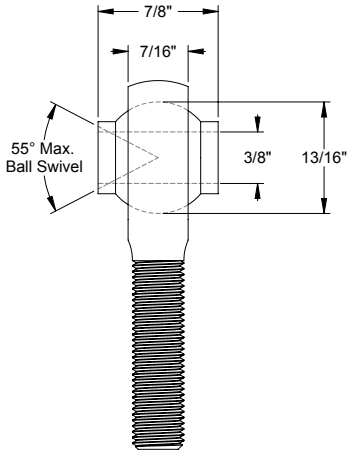
Part Number:

Drawing Number: **1**

Sheet: **1 of 1**

Revision: **A**



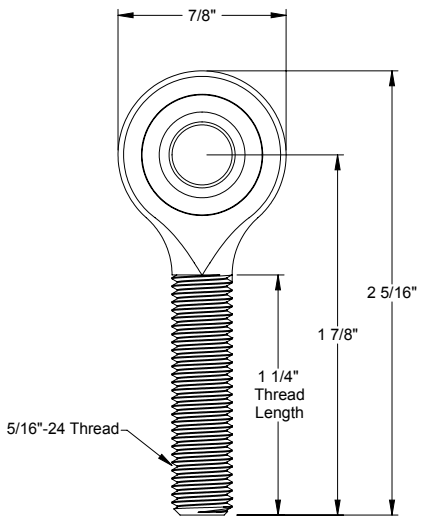


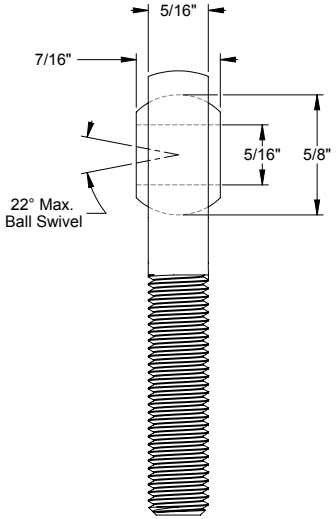
Notes: Heat Treated, Black-Oxide Coated, Alloy Steel Housing
Heat Treated, Chrome-Plated Bearing Steel Ball
PTFE Liner

McMASTER-CARR CAD
http://www.mcmaster.com

PART NUMBER **6960T61**

Right-Hand Thread
Ball Joint Rod End



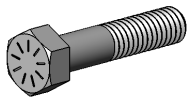


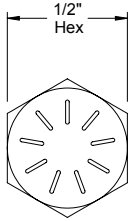
Notes:
Zinc-Plated Steel Housing
Chrome-Plated Steel Ball

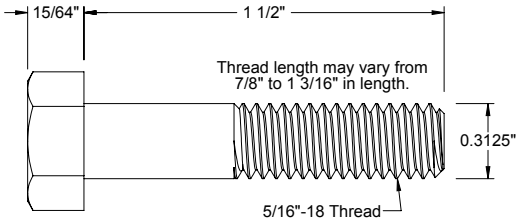
McMASTER-CARR CAD
http://www.mcmaster.com

PART NUMBER **60645K131**

Right-Hand Thread
Ball Joint Rod End



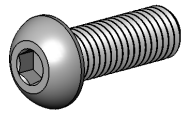


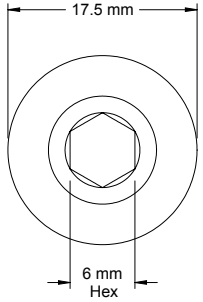


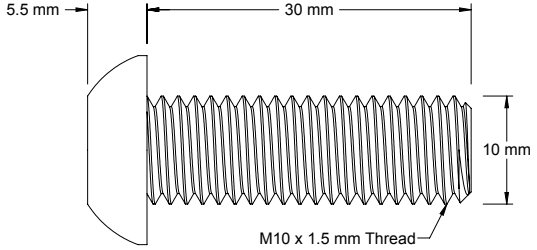
McMASTER-CARR CAD
http://www.mcmaster.com

PART NUMBER **90201A228**

Extreme-Strength Steel
Cap Screw-Grade 9



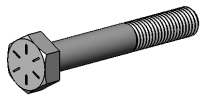


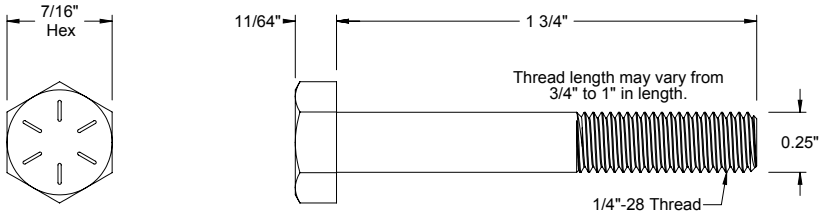


McMASTER-CARR CAD
http://www.mcmaster.com

PART NUMBER **91239A515**

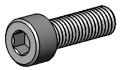
Alloy Steel Button-Head
Socket Cap Screw

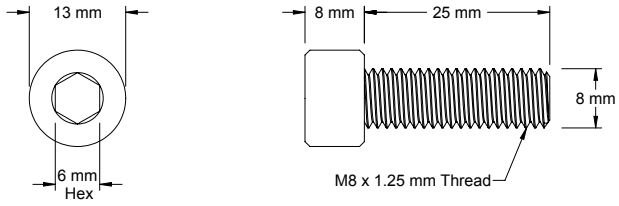




McMASTER-CARR® CAD
http://www.mcmaster.com

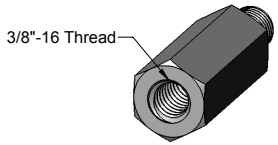
PART NUMBER **91257A567**
High-Strength Steel
Cap Screw-Grade 8

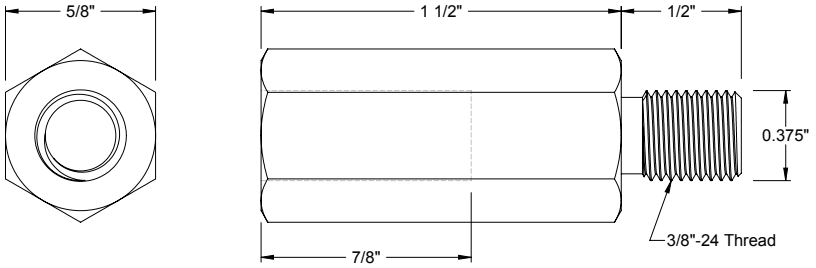




McMASTER-CARR® CAD
http://www.mcmaster.com

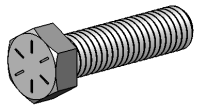
PART NUMBER **91292A148**
Stainless Steel
Socket Head Cap Screw

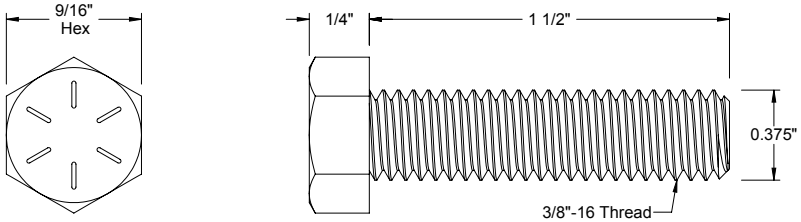




McMASTER-CARR® CAD
http://www.mcmaster.com

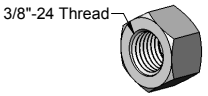
PART NUMBER **92499A298**
Male-Female
Hex Thread Adapter

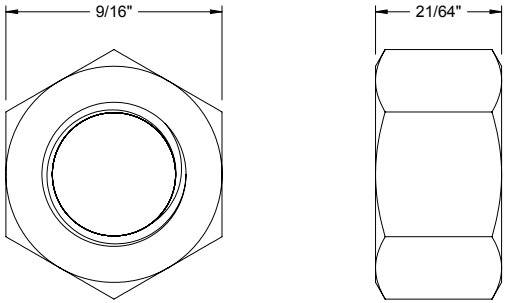




McMASTER-CARR® CAD
http://www.mcmaster.com

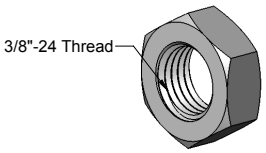
PART NUMBER **92620A628**
High-Strength Steel
Cap Screw-Grade 8

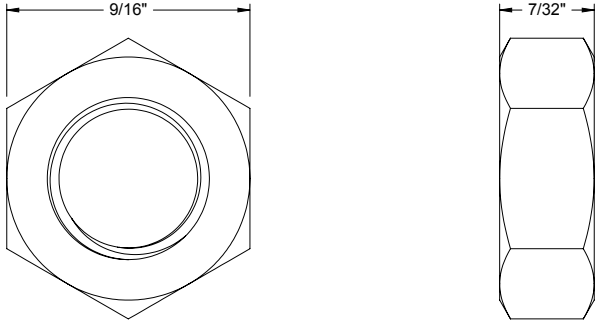




PART NUMBER **90499A815**

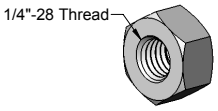
Hex Nut

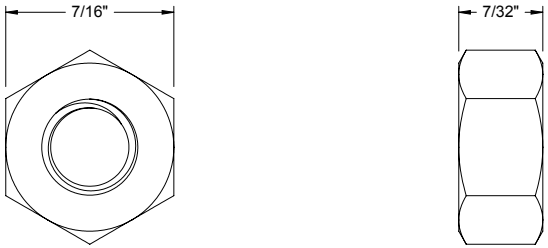




PART NUMBER **94846A515**

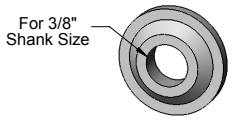
Thin Hex Nut

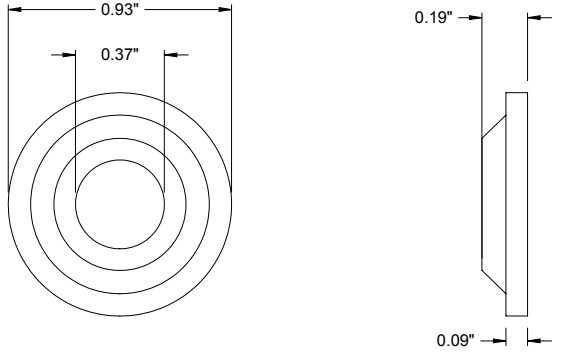




PART NUMBER **95462A505**

Hex Nut





PART NUMBER **95034A300**

Zinc-Plated Steel Washer for Ball Joint Rod Ends