EVS28
KINTEX, Korea, May 3-6, 2015

AutoBikes: Autonomous Electric Bicycles for First and Last-mile Mobility on Demand

Erik Wilhelm[1], Abhishek Gupta[1], Selvasurendhiran Modurpalayam[1], Sirui Wang[2], Brandon Walton[2], Adam Wootton[2], Tim Jeruzalski[2], Andrew Andrade[2], Soh Gim Song[1]
[1]*Singapore University of Technology and Design, 8 Somapah Rd. 487372 Singapore, erikwilhelm@sutd.edu.sg*
[2]*University of Waterloo, 200 University Avenue West, Waterloo, Ontario, N2L 3G1 Canada*

Abstract

Despite significant interest and large-scale deployment in major cities worldwide, urban bicycle sharing systems suffer from significant drawbacks resulting in a reduction in overall impact. Foremost among these problems are the difficulties presented in re-balancing the bicycle fleets to optimally serve demand with bicycle supply in real-time, the cost and footprint of sharing station infrastructure, as well as the substantial costs of damage and theft. The AutoBike technology platform aims to solve rebalancing, damage, kiosk space, and real-time demand management problems by enabling bicycles to cruise autonomously while rider-less, and provide route guidance and rider monitoring while ridden. The AutoBikes are designed to be self-driving when not being ridden, removing the need for traditional rental kiosks thereby saving significant infrastructure and land rental costs. The AutoBikes are able to recognize road boundaries and to avoid collision with both static and dynamic obstacles. They can be automatically rented via RFID cards either on-the-fly, or via pre-booking on a smartphone, and are only marginally more expensive than standard electric bicycles. The work presented in this paper will cover the mechanical, electrical, and control architecture for the AutoBike design. The results of various computer-vision experiments will be presented illustrating success in performing vision-only autonomous vehicle control in both dynamic and static environments through statistical machine learning approaches. Finally, the overall system architecture and business models will be treated.

Keywords: mobility on demand, autonomous vehicle, electric bicycle, computer vision, dynamical system control

1 Introduction

Shared mobility is an old idea whose time has finally arrived. In order to meet the travel demands of a growing urban population, particularly in first and last mile scenarios, large cities are increasingly promoting alternatives to personal cars which often involve cycling. Bicycle sharing systems have been introduced in various forms in over 700 cities around the world, with implementations ranging from free, uncontrolled community bikes to technologically advanced, secure systems [1]–[3]. The features common to most successful systems are:

- Dense vehicle deployment over 10-15 km^2 (300 m average between stations),
- Fully automated locking systems,

- Vehicle tracking using wireless and RFID technologies,
- Real time fleet monitoring,
- Real time user interfaces through web, mobile phone, and/or on site terminals,
- Pricing structure incentivizing short trips to maximize the number of trips.

Concurrently, the growth in the global electric bicycle market is rapid, with much of the demand coming from China and other Asian countries [4]. There have already been several electric bicycle sharing schemes launched, including the one used to motivate this research.

A survey conducted in one of the largest and most successful bike sharing systems called 'Bicing' in Barcelona indicated two major frustrations for users: (a) the difficulty to find an available bike when users want to start their journey and (b) the difficulty to find an empty drop off location for the bike [5].

These underlying frustrations and inconveniences, together with the high cost of fixed infrastructure in urban settings and cost of rebalancing and maintaining a shared mobility fleet has motivated the research outlined in this paper. The innovations which will be presented here include those relating to the stability and steering mechanisms for a bicycle, its low level as well as supervisory, vision, and human-in-the-loop control systems, and the methods used for leaving invisible signals for autonomy. A system-level analysis of the benefits of autonomous rebalancing of fleets will be briefly presented.

2 Mechanical Systems

Enabling a bicycle to ride autonomously poses two major mechanical challenges. Firstly, the bicycle must be stable in all operating modes, and secondly, the bicycle must be both manually and automatically controllable. Various strategies for achieving stability for bicycles based on gyroscopes or inverted pendulums have been presented [6], [7]. In the shared mobility application, these would add unnecessary complexity. For example, use of heavy flywheels for a control-moment gyro would impede manual riding [8], while employing incessant steering to balance the vehicle would encumber navigation in slow-moving traffic [9].

The aforementioned technologies may mature to the point at which they are practical in cost and complexity constrained applications such as the one developed here, but at present the only feasible solution is to separate balancing and steering control functions and implement retractable balancing wheels for stability at lower speeds with an electrically driven steering column as shown in Figure 1. This approach still allows for the future implementation of self-stability control above a threshold velocity with retracted wheels by developing a controller based on an understanding of the gyroscopic and caster effects in the dynamical system as described here [10], which will be the subject of future work.

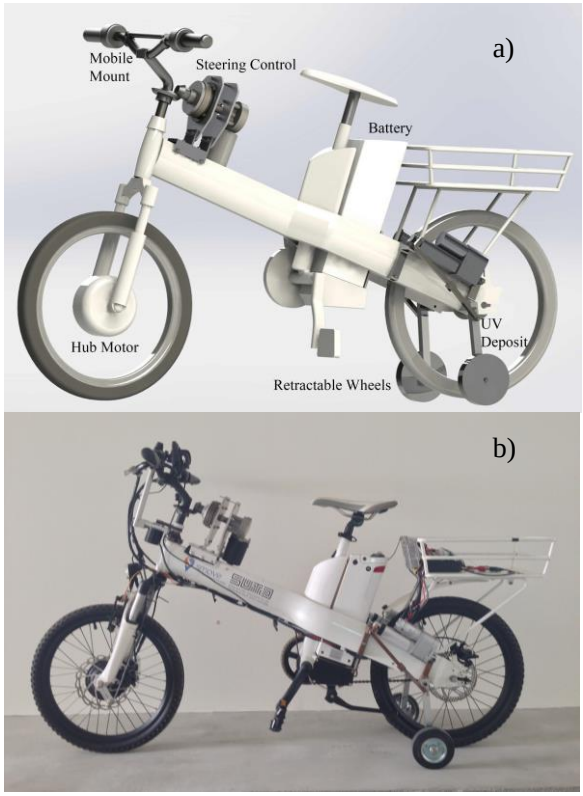


Figure 1: a) CAD model of Auto-bike with steering and wheel retraction assemblies. b) Physical embodiment of autonomous bicycle

2.1 Propulsion and Braking

The bicycle is driven by a 250 W brushless DC motor on its front wheel with a maximum speed of roughly 25 km/h with a 20-inch (0.508 m) diameter tire. The maximum speed during autonomous operation, however, is limited to 4-5 km/h depending on road grade.

Braking while in autonomous mode is achieved using a cantilever braking system distinct from the bicycles' primary disc braking system. An electric linear actuator which can supply a peak force of 23 N is used to operate the brake. The vehicle' stopping ability from its maximum velocity of 4-5 km/h was tested, and the brake was

able to achieve a complete stop with a deceleration of 0.09 G on planar surfaces.

2.2 Steering

The torque required to steer the unmanned bike at standstill was measured to be 6.2 N-m at the steering column. An initial design using a 12 V, 20 W bipolar stepper motor with chain and sprocket transmission having a mechanical advantage of 3.6 failed due to i) its inadequate velocity ratio and ii) the required high input torque from both the stepper motor back-EMF and steering mass for manual control.

A solution was to employ a bevel gear and clutch system. The final design shown in Figure 2 included an electromagnetic (EM) clutch with a bevel gear train (2:1 speed ratio) in place of the chain to make it more compact. The motor is a 12V DC, 60 W worm gear motor, which has a rated speed of 50 rpm and torque of 11.5 N-m.

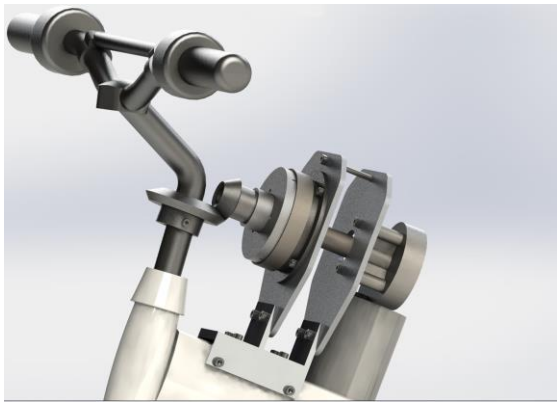


Figure 2: Auto-bike steering assembly

With this configuration the achievable steering accuracy is 12 degrees, resulting in acceptable controllability. The steering speed found to be optimal for most real-world driving scenarios is 30 degree/s. The EM clutch also adds the flexibility of instantaneous engaging/disengaging that is useful when experimenting with freewheeling at high speeds.

2.3 Wheel Retraction

The balancing wheels are design to absorb dynamic loads expected during acceleration, braking or traversing obstacles. A simple triangulated member would be apt for the former while being weaker during tangential loading. The wheels must also be quickly deployed should the vehicle's speed be suddenly reduced.

To satisfy these constraints, a reverse sliding crank mechanism with a linear actuator was first implemented. It caused excessive back-loading on the actuator causing failures in real-world driving and was therefore discarded.

A modified version of an aircraft landing gear mechanism is now employed which compartmentalizes the loading on the wheel and the actuator. A 110 lbs (489 N) max force actuator with 2-inch (0.0508 m) stroke is used in the present design. A bell-crank connected to the actuator achieves total wheel lift within this short stroke and a pivoted connection between this lever and the wheel holding frame ensures locking when wheels are down. Figure 3 shows the wheels when fully retracted and extended.

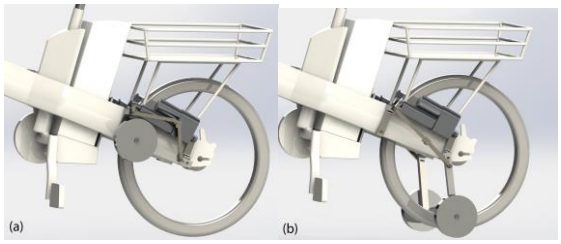


Figure 3: Wheel retraction assembly (a) when extended, (b) when retracted

3 Electrical and Control Systems

The bicycle's control system is comprised of two main embedded systems communicating with one another using a UART serial link. A BeagleBone Black microcontroller connected to a custom motor driver board is responsible for low-level control of the bicycles dynamic behaviour. For example, this controller would receive a speed and heading input, and ensure that the vehicle responds appropriately.

The second controller can be considered the supervisory controller, and consists of a mobile phone running the Android OS with a custom application software that is responsible for maintaining a connection to a web server, GPS route planning, computer vision processing and dynamic model calculations. For example, the supervisory controller would receive a set of waypoints from the dispatch web server, and would be responsible for translating these into speed and heading commands to travel this route.

The low-level control board interprets commands from the Android phone in order to actuate various components of the electrical system that allow the bike to move. It is also responsible for relaying sensor feedback to the phone and maintaining safe operating conditions for the bike by mitigating failure conditions. The overall interface architecture is shown in Figure 4.

Each of the control components will be discussed in more detail in the following sections.

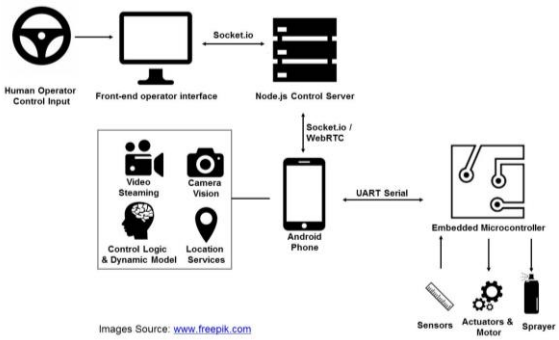
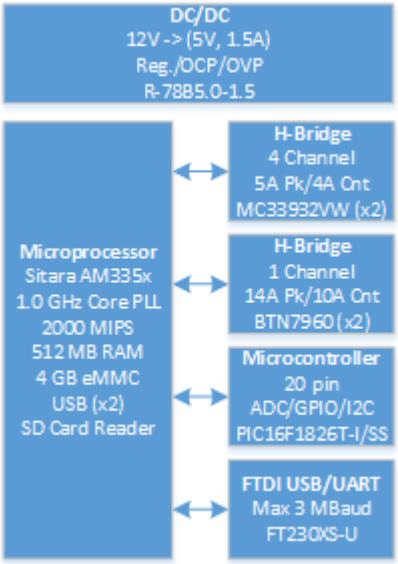


Figure 4: AutoBike interface architecture

3.1 Electrical Systems

To facilitate the power electronic components on the bicycle, a custom driver board was created. The board contains 5 full H-bridge channels, an on-board microcontroller and a USB to UART interface IC. The control board can switch currents up to 10A. The conceptual schematic for the board is shown in Figure 5a, and the physical board in Figure 5b.

a)



b)

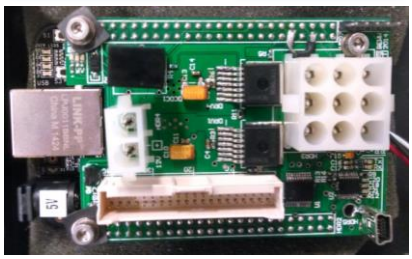


Figure 5: a) The layout of the custom BeagleBone Black electric control shield b) the implemented shield

3.2 Low-level Control Systems

The low-level control system is intended to provide safety critical, real-time sensing and actuation of the bicycle's low-level mechatronic components. These components include linear actuators, and DC motors. Sensors include linear actuator position sensors, wheel speed sensors, steering angle encoder, and a 9-axis inertial measurement unit. The low-level embedded controller is responsible for interfacing with a custom power electronics board which in turn directly drives the DC steering motor, provides a signal for the bicycle drive motor, and linear actuators.

3.2.1 Custom controller

The main controller itself consists of a TI AM335x series processor packaged in the BeagleBone Black. A comprehensive set of custom drivers were written to interface directly with the low-level processor meaning no operating system is employed. This gives exclusive access to all hardware resources available to the processor, most notably the NEON hardware floating point accelerator. Direct processor resource access was chosen to increase the level of determinism of the system and to more easily meet the model time constraints. The software on the low-level controller executes in hard real-time at a rate of 1 kHz.

For mobility on demand applications, the boot time of the system is a key consumer acceptability factor. The choice of running a 'bare-metal' controller allows for boot times of well under 2 seconds from power–on to full functionality.

Given the high reliance of automatically generated code found throughout the automotive industry, bindings were created to allow the low-level controller code to be authored in MATLAB and Simulink. This also allows the controller to be easily re-used to target other applications, and drastically increases the ease of use for implementations on this controller. The interactions between the low-level controller and the other control systems are shown in Figure 6.

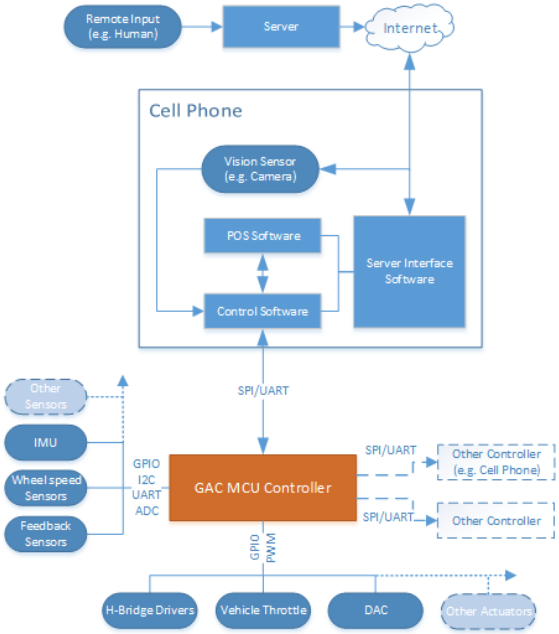


Figure 6: Custom low-level control interactions with overall control system

At each of the 1 kHz software "steps", the current value of each on-board sensor is read into memory by the C++ drivers. The compiled Simulink model code is then advanced one discrete simulation step, and the sensor values are accessed in order to determine appropriate actions. Additionally, one command is read from the UART serial interface. Depending on the nature of the command, appropriate actions will be triggered by the model. For example, a remote-control command may request a certain steering angle, and the model will determine the appropriate direction to turn the steering motor depending on how its currently sensed position relates to the requested angle.

The model will also issue one or more UART response messages to the supervisory controller at each step. These can include a confirmation of entering autonomous mode, indication of a low-level problem such as a non-fatal electrical failure, or state updates that notify the supervisor of the current readings from each sensor.

Despite these sensor updates, the overall control system still operates using a feed-forward model. This means that the supervisory controller (the Android phone) issues control commands based on its current understanding of the situation. Computer vision, GPS-fused location, and learned behaviour are the main factors that influence this decision. The phone does not use the sensor values as part of its understanding. In this way, the low-level controller is trusted to enact the phone's requested behaviour based on its current understanding of the state of its sensors and the physical characteristics of the actuation system it is driving.

By avoiding the need for up-to-date sensor feedback on the supervisor controller, possible saturation of the UART communication link is avoided. Additionally, timing issues caused by delays in sending the latest sensor values to the phone do not need to be considered. The design also allows the supervisory controller to more easily used in a different vehicle system with the same UART interface, because the exact physical nature of the vehicle is kept separate from the high-level control logic.

3.3 Autonomous Control Systems

The AutoBike autonomous control system is managed by the Android device attached in front of the handlebars on the bike, which acts as the supervisory controller. This system is responsible for the communication between the web server and bike controller and performs autonomous route planning based on server waypoints and computer vision inputs. A data communications link is maintained with the external server through the use of asynchronous Socket.io operations, and enables a remote operator to issue commands to the Auto Bike from a front-end web interface. These can include tele-operation commands, or signals to switch between autonomous and tele-operation modes. The link also allows the server to be kept up-to-date on the bike's current position and state, information which can be relayed to the human operator.

The status of the UART communication link between the phone and embedded controller is monitored by both parties. A periodic health check is sent back and forth to indicate continuing operation. If this check is disrupted for any reason, the embedded controller will immediately enter a safe failure mode wherein the bike will be navigated to an immobile, safe state, while the phone will notify the server and human operator of the failure and attempt to re-establish contact.

The route planning is a mix of several components which provide a lightweight and effective system. These include a vision system, GPS localization fused with sensor inputs, and learned responses derived from the algorithm observing human operator behaviour. High level route planning is performed on the server using GPS data and a waypoint system utilising spline interpolation between the GPS waypoints.

Additional information is required for driving as the GPS location data returned by the Android device is unable to provide a sufficiently accurate position. This data comes from the vision system, pre-recorded human tele-operation ("learned behaviour"), and filtered dead-reckoning using dynamic model simulation.

3.3.1 Vision System

The autonomous control system is primarily managed by a vision system running on the Android device. The vision system is tasked with determining a safe operating envelope (SOE) for the bike which describes the areas which the bike can travel. Utilising the camera from the Android device and performing image processing techniques it is possible to identify the path and potential obstacles in front of the bike in real-time. There have been several successful studies utilising machine learning algorithms in order to navigate roads and paths [11]. The SOE determination is limited as the Android device only has one camera, which leads to only a small SOE which can be determined limited by the field of view of the camera. The three main steps of the process are shown in Figure 7.

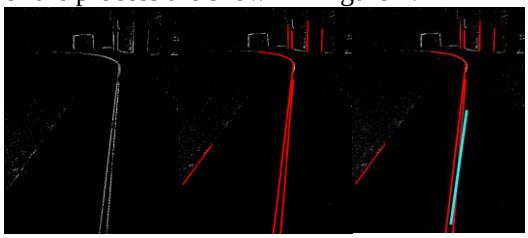


Figure 7: a) video frame after Canny edge detection b) video frame after Hough line detection, c) representative image of a RANSAC fitting (not computed)

A low resolution greyscale image is captured from the Android device. 320 by 240 pixels were sufficient for the AutoBike. OpenCV functions are thereafter used for processing the image data. A high-pass filtering operation is performed to identify the white regions of the scene, which is then processed through a Canny edge detection filter and probabilistic Hough line detection filter. The outputs of these filters highlight the line segments which are of interest to the AutoBike which are then fed into a modified RANSAC algorithm with rules limiting certain slopes and positions for each line segment. This finds the optimal line to follow while ignoring irrelevant horizontal lines and others from the surroundings which are not part of the path to travel.

After finding the optimal line to follow it is used in a decision tree voting structure which determines which direction the bike needs to move. Based on rules from the positioning of the line in the image space and the angles associated with said lines, it determines the optimal angle which to turn the AutoBike. In the case that there is not enough information to decide which way the AutoBike should turn it falls back into a safety mode where another control strategy takes over, where either teleoperation or autonomy based on learned human actions during previous failure events is applied.

3.3.2 Teleoperation and Path Marking

At any time the AutoBike can be directly controlled by the human operator through tele-operation. In this mode the scene in front of the bike is streamed through the webserver to the driver with minimal latency. The driver can control the system through a joystick control which will allow the AutoBike to navigate through difficult situations. In case of the vision system unable to determine the correct direction to travel, the system prompts for human intervention and will record the actions performed by the driver. These recorded actions then go on to help train the AutoBike for later trials. If the bike experiences issues in the same location repeatedly then the system will use the previously recorded actions to navigate the area. These recordings are stored on the server and the route is transmitted to the controller once the vision system has failed.

Due to limitations with the Android operating system, the camera peripheral cannot be accessed by both WebRTC and the computer vision algorithms concurrently. A consequence of this is that while the bike is in autonomous mode, no video output can be streamed using WebRTC. To counter this problem, a system was devised wherein a rapid sequence of highly compressed JPEG images output from the computer vision algorithms could be sent in place of the WebRTC stream. As remote control operations are not used in this state, a lower framerate and higher compression ratio can be used to maintain the same level of bandwidth consumption while still allowing remote operators to observe the general behaviour and state of the bike while it is in autonomous mode.

An important aspect of the technology development is the ability of the vehicle to deposit markings which the fleet of bicycles can use to decide on their location and actions at a given time. For example, the bicycle operation may

benefit significantly from the ability to localize based on relative position from a known set of symbols on the pavement. Or perhaps the vehicles mark areas where human tele-operators may need to consistently step in to navigate difficult intersections of other challenging environments.

To accomplish this behaviour, a paint containing pigments which phosphoresce under UV light is deposited automatically, or under the discretion of a human operator, through a set of selectable stencils to encode information. The information is then sensed using a second image sensor mounted in a dark container with a UV light source. Initial tests have been performed using the system shown in Figure 8, and results are positive. Video demonstration of this feature is shown via the link in the conclusions section.

Figure 8: Path marking system displaying the aerosol delivery of UV-luminescent pigments

3.4 Server Control Systems

A web server and communications solution is developed for monitoring and controlling the autonomous package over existing IP-based networks. The Node.js-based server platform is easily scalable due to its simplicity, which allows for management of a large bicycle fleet in the future. To ensure message integrity, the server actively checks transmission latency between all clients connected. For clients, messages received after timeout or with bad checksums are discarded, and relevant fail-safe mechanisms invoked.

The server control system enables full-duplex data as well as real-time Audio/Video transmission between clients. Data channel is used for telemetry and remote control, while the multimedia channel provides feedback to manual remote operators. Optimizations such as data compression are used to minimize transmission delay. Furthermore, Audio/Video streaming from AutoBikes to operator consoles utilizes point-to-point WebRTC technology, which decreases delay as it bypasses the central server.

During development, a gaming steering wheel is used as the Human Interface Device for throttle and steering control. The operator console streams inputs from the gaming steering wheel to the server, which then relays the information to the AutoBike.

The interface also provides controls and information displays to aid in remote navigation of the vehicle. As seen in Figure 9, a map with the current location of the vehicle, as well as the remote video feed, are displayed. Buttons to switch between autonomous, remote-control and emergency stop modes are available, as well as a visual indication of the mode the vehicle is currently in. There is also a message console that allows the vehicle to issue textual debug and status messages to the human operator.

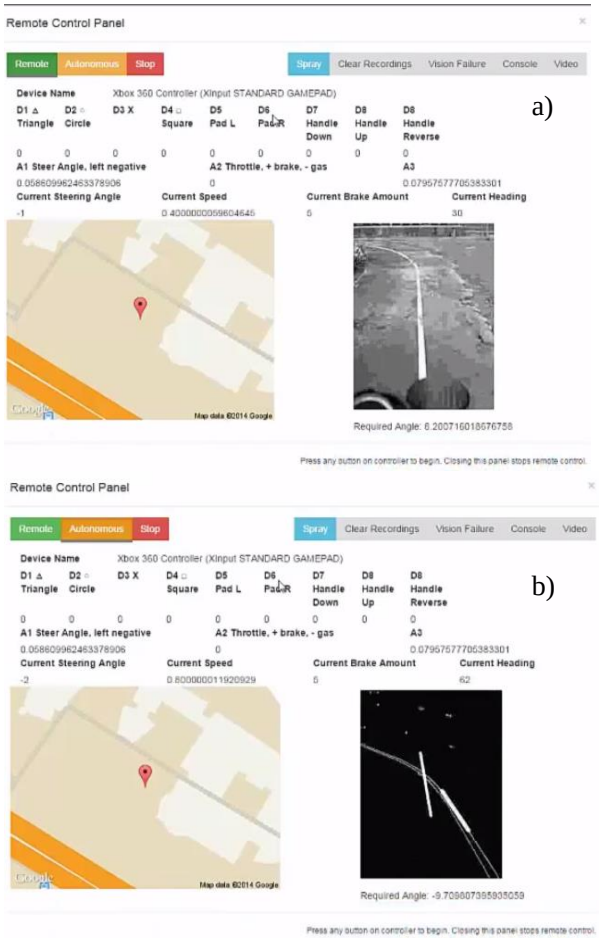


Figure 9: a) in remote teleoperation mode, the driver sees a heads up display of the vehicle's location; b) in autonomous mode, the vehicle identifies a safe operation zone to travel between waypoints

Internal to the autonomous package, a USB-based UART (Universal Asynchronous Receiver-

Transmitter) communications solution is also developed for interfaces between the Android smartphone and Embedded Controller. The custom external communications solution features a full-duplex data channel with Cyclic Redundancy Check checksum support, ensuring a high-speed reliable delivery of binary data. A messaging protocol governs the specific composition of messages exchanged in this channel.

4 Rebalancing and Business Models

A Mobility-on-Demand (MOD) shared vehicle system should provide convenient availability of vehicles to users upon demand. Previous attempts have been made to solve this problem through (a) improving the redistribution of bikes from full to empty stations using large logistics vehicle and (b) informing users in advance about the best places to pick up or leave bikes and using price signals to try and influence behaviour.

The vehicle routing problem is one of the most studied combinatorial optimization problems, and has been successfully applied to maritime logistics, commercial trucks and vehicle sharing operations [12]–[15]. Specific to bike sharing systems, rebalancing approaches have been taken using data mining and exploratory data analysis, static repositioning, and soft computing [16]–[18]. Many studies have also been conducted to understand bike usage patterns and predict user demand including an ongoing data science competition [19].

These approaches do not solve the underlying frustration that users must travel a distance from their start or end location which adds inconvenience to the trip. A system where robotic self-driving vehicles rebalance themselves to ensure acceptable coverage area and most importantly have the ability travel to users upon demand, thus reducing overall the trip time and inconvenience for users.

4.1 Dynamic System Model

A simple dynamic model was created to contrast the both traditional station based bike sharing model with an autonomous model. The model calculates the average trip time for a number of trips given a system coverage area and uniform demand. Ratios for the number of bikes, docking slots and stations are set based on empirical data from various public data on the traditional station based models. Bicycles are not available for an unlimited time. While fleet rebalancing is present to deal with general usage trends based on historic demand, the dynamic model simulates situations where some stations do not have bikes present due to real time inventory.

An autonomous bike sharing system with an equal number of total vehicles is contrasted with a standard station-based MoD system using a simulation model. The simple system model is show in Figure 10. The trip time for the autonomous system is the sum of the time waiting for the bicycle to arrive where it is needed and the time taken to cycle to the destination. The trip time for the station-based system is the sum of the walking time to the station, time to cycle to the next open station, and then the time to walk from station to destination, where walking time is Euclidian distance, and cycling and autonomous driving time is calculated using Manhattan distance.

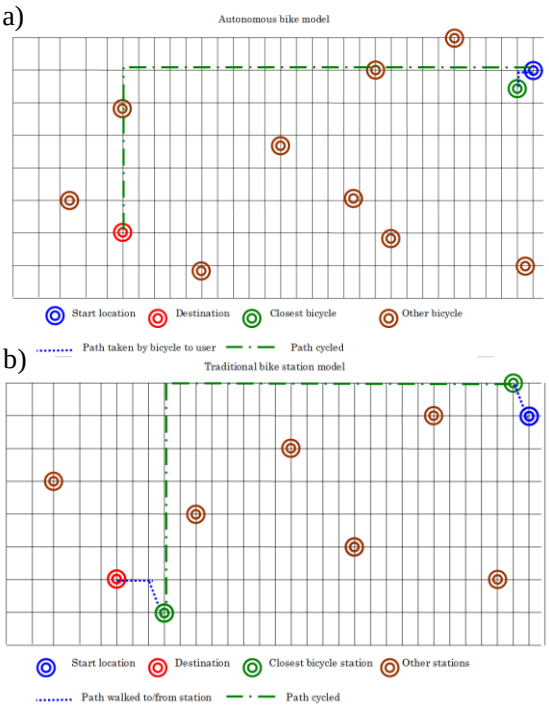


Figure 10: a) Autonomous rebalancing system diagram; b) Station-based Mobility-on-Demand systems

The results of the simulation are shown in Figure 11. The largest contributor to trip time is walking which is fairly insignificant in smaller coverage areas. For short distances and coverage area, both systems provide adequate service and feasibility in usage, but as the system coverage area increases, autonomous bike sharing models make a larger

improvement due to the reduction in walking time. This makes such a system feasible both in densely populated areas and in areas where a traditional bike sharing model would be impractical.

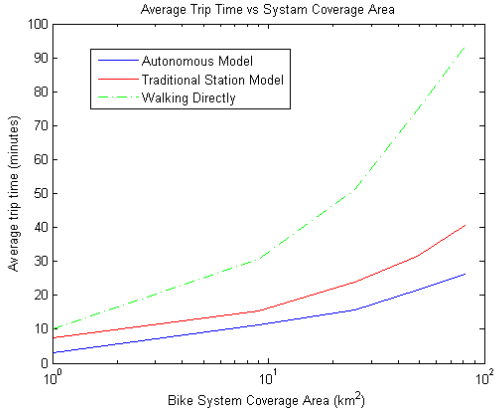


Figure 11: The benefit of the autonomous MoD rebalancing model increases as the coverage area increases

5 Summary and Future Work

This paper has outlined the implementation of an autonomous bicycle for self-rebalancing Mobility on Demand fleets. All of the functionality described here has been validated and tested. The results of the test can be seen in the video associated with this work: http://people.sutd.edu.sg/~erikwilhelm/autobikevideo/

Acknowledgments

The authors would like to thank the MIT SMART program, the SUTD-ZJU Collaboration Program, and the SUTD-MIT International Design Centre for financial support in undertaking this research project, and acknowledge its industry partner Smove for donating its shared fleet of bicycles.

References

[1] P. DeMaio, "Bike-sharing: History, impacts, models of provision, and future," *World Transit Res.*, vol. 14, no. 4, pp. 41–56, Jan. 2009.

[2] Jemilah Magnusson, "The Bike-Share Planning Guide," *Institute for Transportation and Development Policy*, 2013. [Online]. Available: https://www.itdp.org/the-bike-share-planning-guide-2/. [Accessed: 22-Jan-2015].

[3] Peter Midgley, "The Role of Smart Bike-sharing Systems in Urban Mobility," *Journeys*, vol. 2, pp. 23–31, 2009.

[4] Statista, "E-bikes - worldwide sales by region 2018 | Forecast," *Statista*, 2014. [Online]. Available: http://www.statista.com/statistics/255658/worldwide-sales-of-electric-bicycles-by-region/. [Accessed: 29-Jan-2015].

[5] A. Kaltenbrunner, R. Meza, J. Grivolla, J. Codina, and R. Banchs, "Urban cycles and mobility patterns: Exploring and predicting trends in a bicycle-based public transport system," *Pervasive Mob. Comput.*, vol. 6, no. 4, pp. 455–466, Aug. 2010.

[6] A. V. Beznos, A. M. Formal'sky, E. V. Gurfinkel, D. N. Jicharev, A. V. Lensky, K. V. Savitsky, and L. S. Tchesalin, "Control of autonomous motion of two-wheel bicycle with gyroscopic stabilisation," in *1998 IEEE International Conference on Robotics and Automation, 1998. Proceedings*, 1998, vol. 3, pp. 2670–2675 vol.3.

[7] S. Lee and W. Ham, "Self stabilizing strategy in tracking control of unmanned electric bicycle with mass balance," in *IEEE/RSJ International Conference on Intelligent Robots and Systems, 2002*, 2002, vol. 3, pp. 2200–2205 vol.3.

[8] P. Y. Lam, "Gyroscopic stabilization of a kid-size bicycle," in *2011 IEEE 5th International Conference on Cybernetics and Intelligent Systems (CIS)*, 2011, pp. 247–252.

[9] Y. Tanaka and T. Murakami, "Self sustaining bicycle robot with steering controller," in *The 8th IEEE International Workshop on Advanced Motion Control, 2004. AMC '04*, 2004, pp. 193–197.

[10] J.D.G. Kooijman, J.P.Meijaard, Jim M. Papadopoulos, Andy Ruina, and A.L. Schwab, "A bicycle can be self-stable without gyroscopic or caster effects," *Science*, vol. 332, no. 6027, pp. 339–342, Apr. 2011.

[11] J. Michels, A. Saxena, and A. Y. Ng, "High Speed Obstacle Avoidance Using Monocular Vision and Reinforcement Learning," in *Proceedings of the 22Nd International Conference on Machine Learning*, New York, NY, USA, 2005, pp. 593–600.

[12] Bruce Golden, L., S. Raghavan, and E. Wasil, *The Vehicle Routing Problem: Latest Advances and New Challenges*, vol. 43. Springer, 2008.

[13] E. M. Claessens, "Optimization procedures in maritime fleet management," *Marit. Policy Manag.*, vol. 14, no. 1, pp. 27–48, Jan. 1987.

[14] H. Mahmassani, Y. Kim, and P. Jaillet, "Local Optimization Approaches to Solve Dynamic Commercial Fleet Management Problems," *Transp. Res. Rec. J. Transp. Res. Board*, vol. 1733, pp. 71–79, Jan. 2000.

[15] R. Nair and E. Miller-Hooks, "Fleet Management for Vehicle Sharing Operations," *Transp. Sci.*, vol. 45, no. 4, pp. 524–540, Nov. 2011.

[16] P. Vogel, T. Greiser, and D. C. Mattfeld, "Understanding Bike-Sharing Systems using Data Mining: Exploring Activity Patterns," *Procedia - Soc. Behav. Sci.*, vol. 20, pp. 514–523, 2011.

[17] T. Raviv, M. Tzur, and I. A. Forma, "Static repositioning in a bike-sharing system: models and solution approaches," *EURO J. Transp. Logist.*, vol. 2, no. 3, pp. 187–229, Aug. 2013.
[18] L. Caggiani and M. Ottomanelli, "A Modular Soft Computing based Method for Vehicles Repositioning in Bike-sharing Systems," *Procedia - Soc. Behav. Sci.*, vol. 54, pp. 675–684, Oct. 2012.
[19] Kaggle, "Bike Sharing Demand," 2014. [Online]. Available: https://www.kaggle.com/c/bike-sharing-demand. [Accessed: 22-Jan-2015].

Authors

Erik is an Assistant Professor at the Singapore University of Technology and Design (SUTD). He received his Honours BASc and MASc in Chemical Engineering from the University of Waterloo in 2005 and 2007 respectively, and his PhD from the ETH Zurich with Technology Assessment group at PSI in 2011. He performed post-doctoral research at the Massachusetts Institute of Technology before joining the SUTD.

Abhishek is a Research Engineer in the Motion, Energy, Control lab at the SUTD with expertise and research interests in embedded systems design. He has a Bachelor Degree from the Vellore Institute of Technology, and a Master degree from the Nanyang Technological University NTU.

Selva is a Research Associate at the Singapore University of Technology and Design. He received his BE in Automobile Engineering from Anna University, Chennai (India) and his MSc in Mechanical Engineering from National University of Singapore (Singapore). He is currently working on mechanical aspects of autonomous vehicles and energy storage technologies.

Sirui is a former research assistant at the SUTD, focussing on the real time web interface for the AutoBike. He is currently completing his Honours BASc at the University of Waterloo.

Brandon Walton is a senior student at the University of Waterloo in Waterloo, Canada. He is currently pursuing a bachelor's degree in science specializing in applied physics. His areas of interest include hybrid electric vehicles, control strategy development and the modelling/simulation of physical phenomena. He is currently the controls lead for the University of Waterloo advanced vehicle technology competition team.

Adam Wootton is a research assistant at the Singapore University of Technology and Design. He is currently completing his BASc in Systems Design Engineering from the University of Waterloo. Currently, he is working on software development for autonomous vehicles.

Tim is a former research assistant at the SUTD, focussing computer vision systems for robots. He is currently completing his Honours BASc at the University of Waterloo.

Andrew is an entrepreneur & AI practitioner on a mission to design, develop and deploy products that revolutionize the world. Studying Mechatronics Engineering at the University of Waterloo, his research interests are in artificial intelligence, robotics and man-machine symbiosis. When he is not building technology, Andrew is travelling the world and climbing mountains.

Gim Song is an assistant professor in Singapore University of Technology and Design. He obtained his MSc and PhD degree from University of California, Irvine (UCI), where he researched and developed innovative computer aided design software for the invention of articulated robotic system. His research focus is on mechanical system design and robotics.