Road Curvature and Camera Parameters for Autonomous Navigation

Joshua G. Send
Trinity Hall

UNIVERSITY OF CAMBRIDGE

A dissertation submitted to the University of Cambridge in partial fulfilment of the requirements for the degree of Master of Philosophy in Advanced Computer Science

University of Cambridge
Computer Laboratory
William Gates Building
15 JJ Thomson Avenue
Cambridge CB3 0FD
United Kingdom

Email: js2173@cam.ac.uk

May 18, 2018

Declaration

I Joshua G. Send of Trinity Hall, being a candidate for the M.Phil in Advanced Computer Science, hereby declare that this report and the work described in it are my own work, unaided except as may be specified below, and that the report does not contain material that has already been used to any substantial extent for a comparable purpose.

Total word count: TODO

Signed:

Date:

Abstract

This is the abstract. Write a summary of the whole thing. Make sure it fits in one page.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Vision

In the future, private vehicles in urban environments have been made redundant by an efficient fleet of autonomous vehicles, ferrying passengers on demand between locations in the city. The city's pool of vehicles consists of vehicles from various manufacturers, but are managed by a public-private partnership, much like a utility, and seen as an essential part of the city infrastructure fulfilling the needs of its citizens.

The utility manages not only a large selection of vehicles, but also the infrastructure that the autonomous vehicles run on: a network of cameras, placed along roads to facilitate vehicle navigation. At this point in time the hardware requirements of object detection algorithms are miniaturized, and are highly accurate, able to identify pedestrians and vehicles to within a few centimeters. These locations are transmitted to nearby vehicles. As a result, GPS, radar, and expensive LIDAR systems have become largely redundant on the autonomous vehicles.

With this infrastructure, autonomy becomes cheaper to implement, and more accessible. Vehicle manufacturers work with the utility to keep the cameras functional and up to date. City GPS canyons are avoided, and autonomy is enabled on everything from bicycles, to small delivery robots, to standard vehicles – all by shifting the sensing task from the individual user to the infrastructure.

1.2 Motivation

The core task of an autonomous vehicle is to navigate correctly and safely from a start to a destination. This process can be split into a variety of subtasks consisting of path planning, localization, and safe navigation.

Current autonomous vehicle projects use a large set of sensors, combined with maps, to solve each of those complex tasks. This comes at a cost: current Level 3 TODO REFERENCE autonomy attempts for example use LIDAR, which provides rich data to perform self-localization and dynamic obstacle avoidance, but also costs around $75,000 [13]. The computational power required to fuse and process data from wheel odometry, GPS, inertial measurement units (IMUs), cameras, RADAR, LIDAR is also substantial and implies only large platforms can be used.

I propose moving the localization and navigation sensing tasks from vehicles to the infrastructure, which is particularly applicable in urban environments. I examine a static environment containing a single vehicle, with dynamic components as a future research possibility.

The basis of most localization techniques are global navigation satellite systems, such as GPS and its augmentations like real time kinematic (RTK) [1]. The US government reports a global 95% accuracy of 0.715 meters [9]. However, GPS errors are introduced in urban environments – according to Miura et al., even with the authors' proposed correction steps, the mean localization error was around 5.2 meters. In contrast, most authors agree autonomous vehicles can tolerate 0.2 to 0.35 meters lateral error [12][7][4].

To achieve these error bounds, current techniques may use computationally heavy visual odometry [7] very high precision maps [4], or expensive LIDAR to localize lane markings very accurately [6].

In this exploratory work, I show that if certain installation and algorithmic constraints can be met, cameras mounted at side of roads can provide the required localization accuracy without reliance on GPS, LIDAR, or RADAR. Wheel odometry is used to navigate between updates received from the infrastructure. I then examine the task of optimal camera placements and configuration for localization and navigational performance.

1.3 Contributions

The practical questions answered by this work are the following:

1. How accurate can offboard (ie. in the environment) cameras be for localization tasks?
2. In which direction should the camera be facing and what should its properties be (for instance, field of view) to optimally aid vehicle localization and navigation, and how is related to road curvature?
3. How should a set of cameras be placed along a path to optimize navigational performance?

The first question relates to feasibility of camera-based localization, and is addressed in Chapter 5.

The second and third questions examine the optimal camera placement task for localization and navigational performance. This is related to past work on landmark placement for robotic navigation, where the robots observe landmarks fixed to the environment. Here, the use case is inverted to observe the robots as they move through the environment. Thus, it is also closely related to previous work on surveillance networks.

While literature on surveillance using cameras has explored camera placement, none has allowed more degrees of freedom than simply adjusting the tilt (pitch) and choosing one of a limited set of positions. Further, no exploration of properties of cameras has been performed - for instance, the impact of allowing a larger field of view in the observation task. Lastly, surveillance tasks normally seek to maximize different objective functions than robotic navigational performance.

Finally, throughout this work I examine the impact of the environment, specifically road curvature, on the various tasks presented. To my knowldge, this variable has not been analysed previously.

In summary, I make the following contributions in this work

- An error model for vehicle locazation using infrastructure-mounted cameras, given an installation uncertainty and estimated vehicle position
- A feasibility analysis for using cameras for localization, using the developed error model
- An analysis of placement of a single camera camera along roads, and the relationship to road curvature
- An analysis of placement of sets of cameras along roads, and the relationship to road curvature

1.4 Overview

Chapter 2 reviews relevant literature in the fields of localization, landmark placement, and surveillance. Chapter 3 discusses required background and the simulation used for analysis.

Chapter 5 develops the error model for cameras used in simulation, and examines the feasibility of the proposed approach, while 6 optimizes single and multiple camera placements in various environments. This is followed by concluding remarks.

Chapter 2

Related Work

* Talk about past work with other sensor types, good/bad

This dissertation touches on a large body of literature ranging from control theory to surveillance optimisation. I focus on past work in surveillance, and...

Chapter 3

Design and Implementation

3.1 Approach

– TODO – rewrite block of text below these section headings into subsections Also touch the items outlined below

3.1.1 Sensors

talk about possible sensor types. camera, why it's useful camera versus passive tags versus tech like UWB, pros/cons.

Here only focus on x,y updates from camera. Pose is possible, cite literature on pose estimation on a variety of tasks.

3.1.2 Vehicle Model and Navigation

* Planar Z = 0 assumption, restrction to x,y,theta * Kalman Filter * odometry model, non-holonomic vehicles * influence of vehicle and noise model on results TODO (=¿ discussion/conclusion?)

3.1.3 Objective Functions

* Localization versus Navigation * Objection functions used in the past (=¿ related work? Or into camera placement chapter?)

3.1.4 Implementation Overview

* simulation – why

* Gaezebo, ROS

* Quality of simulation (+ use of gazebo/ros)

* Quality of results dependent on camera model, error bounds,

* Camera model and relation to kalman filter and navigation

* Parameters optimized for at different points and how this answers the above questions

* TODO decide if totally dropping bits on camera model

The experimentation carried out is performed in simulation. This is partly because of the difficulty of using a real vehicle and a test track, but also since simulation allows quickly evaluating different road curvatures, camera parameters, and camera placements.

In this dissertation, an Ackermann vehicle **TODO CITATION** model in Gazebo **TODO CITATION** was used. The Robotics Operating System (ROS) was used for all control and navigation tasks, while cameras are modelled as separate modules.

Since the results discussed in SECTION TODO depend strongly on the quality of simulation, effort is put into accurate models. Most of the robotic stack developed is based on industry standard tools (e.g. Gazebo and ROS), using realistic control and localization algorithms, though some simplifications are discussed in SECTION TODO.

* TODO * discuss mathematical model of inaccuracy versus real simulated results * impact of lens model

Evaluation of camera parameters is also strongly dependent on the realism of the camera model. Notably, when examining wide field-of-view cameras, the standard pinhole perspective camera can break down. TODO analyses a variety of camera models that can extend all the way to fisheye lenses,

This dissertation fixes camera focal length and resolution, but allows the horizontal and vertical field of view to vary. Additionally, in any individual scenario camera placement is fixed at a certain height, as well as locations along the edge of roads. Camera roll (rotation about the optical axis) is now allowed to vary, while the yaw (rotation about the Z “up” axis) and pitch (tilt) can be adjusted.

*** ROLL this into implementation! ***

Chapter 4

Background

**** talk about coorindate systems somewhere * Roll, pitch yaw * how rotations were implemented (quaternions or matrices)

4.1 Robotics Operating System and Gazebo Simulator

* brief!

4.2 Camera Models

* Discussed further

* TODO skip?

4.3 Extended Kalman Filter

* be sure to write out maths

* discuss use of covariance matrix (x, y, theta) as error function

4.3.1 Model

* discuss alternative (velocity) models

4.4 Navigation and Controls

4.4.1 Curve Representations

* discuss various ways of representing curves

* eg spline, linear approximations, dubins (what I've done), euler

* limitations

* not smooth, more complex, don't generalize well to 3D (This is a bad one!)

* advantages

* analytically easy to solve crosstrack distance, relatively accurate

* less costly than splines

* easy to look up time corresponding to any point in space (related to ease of crosstrack distance)

* unit testing for correctness - 3D geometry easy to mess up

4.4.2 Controllers

* PID, advantages and disadvantages

* Alternative controllers – use of Stanford's steering controller and homebrew velocity controller.

4.5 Submodularity

* TODO – leave to end to see if have time to explore this

Chapter 5

Camera Model

The analysis presented in the next chapter is based on the navigaton of an autonomous vehicle guided only by its own wheel odometry. This implies that errors accumulate over time, and occasional updates need to be provided to reduce the positional uncertainty of the robot. The sensor selected to provide this information is the camera, which is already widely deployed in modern city infrustructure.

The format of the update provided to the vehicle is a location, along with an uncertainty in the form of covariance. This chapter develops the simulated camera which is used to report vehicle location and uncertainty.

5.1 Camera Projection

——— Cut or trim down massively ————

One of the parameters that will be optimized against is the camera field of view. The standard camera model, referred to as a pinhole camera, is generally accepted to model real cameras closely. However, the pinhole model breaks down (ie. looks distorted) when approaching a very wide field of view. The literature varies on where this point is – some sources claim 70 degrees [5], others 90 degrees [14], or even 120 degrees as acceptable. Real ultra-wide-angle cameras are difficult to obtain with more than 114 degrees field of view and usually discard some corner information to map curvilinear distortions back onto straight lines (ie. emulate perspective projection from a fisheye model).

There are several alternative models capable of modeling very wide fields of view: stereographic, equidistant, and equisolid projections are common [2].

In order to discusse their merits, I describe the pinhole model, along with stereographic and equidistant models below.

5.1.1 Generic Camera Model

——— Cut or trim down massively ———

We first begin by assuming the camera is located at the origin, with the principal axis (ie. direction vector) aligned with the $+Z$ axis. Please refer to Figure 5.1 for the following description.

World coordinate P is mapped onto a pixel plane, located at distance f from the world center, via a mapping into spherical coordinates. θ is given as the angle between the principal axis $(+Z)$, and the point P. By projecting P through the origin onto the pixel plane, the angle ϕ can be determined. Lastly, to determine the (x, y) position in the pixel plane, the radius from the origin r is computed as a function of the angle θ. The generic model of Kannala *et al.*shows how the function $r(\theta)$ defines the various models:

$$r = ftan(\theta) \qquad \text{(pinhole perspective projection)} \qquad (5.1a)$$

$$r = 2ftan(\theta/2) \qquad \text{(stereographic projection)} \qquad (5.1b)$$

$$r = f\theta \qquad \text{(equidistance projection)} \qquad (5.1c)$$

In this formulation, we can see that the difference between models is purely the mapping from real-world angle to the camera principal vector. Also clearly evident is that the theoretical limit of the pinhole model lies at 180°

field of view, meaning ±90 in any direction, which is when tan diverges. On the other hand, stereographic projection can handle up to ±180 degrees in any direction before diverging.

5.1.2 Tradeoffs

——— Cut or trim down massively ———

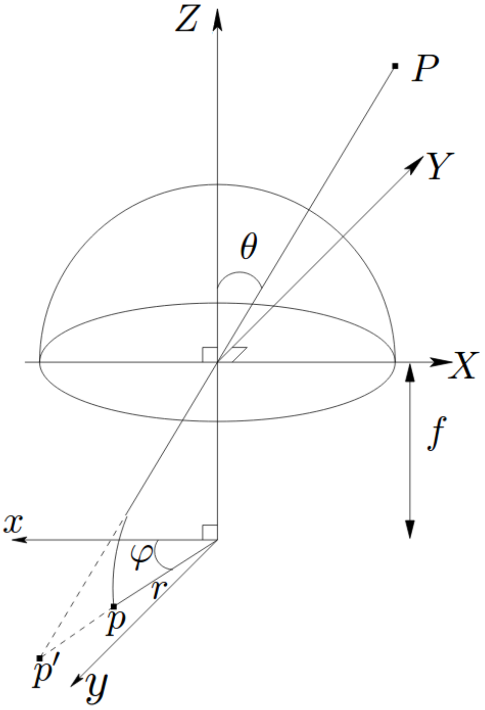

Figure 5.1: The generic model given by Kannala *et al.* [2].

Since I am to simulate localizing a vehicle's world position from the pixels on the camera plane, and derive an associated localization error, I examine a bound on localization accuracy in the form of ground area covered per pixel. That is, in the worst case a vehicle will be contained within 1 pixel, so the ground area covered by each pixel is an upper bound for localization error.

The literature suggests that a good camera model should closely approximate real cameras, and have useful mathematical properties [14]. Pinhole perspective projections preserve straight lines, but not circular shapes as field of view widens. On the other hand, steregraphic models preserve circularity everywhere but not straight lines. The choice here will depend on the type of algorithm selected.

5.1.3 Conclusion

——— Cut or trim down massively ————

In order to be able to analyze the use of very wide fields of view, I implemented this unified model as a ray tracing module with a fixed focal length of 0.1 meters, which according to [10] is sometimes used in traffic cameras. Besides being able to select the specific model to use, the resolution is allowed to vary, as well as the horizontal and vertical maximum field of view.

However, while this section has presented alternative camera models, I will mostly continue to use the pinhole model, unless the field of view is pushed beyond 90 degrees in either direction, in order to conform with the type of camera most likely to be used in practice.

5.2 Localization and Error

As previously mentioned, the cameras deployed in the infrastructure will send an estimated real world position to the vehicle as it travels through the camera's catchment area, reducing its positional uncertainty. This requires deriving an accurate simulation how real-world cameras would perform this task.

5.2.1 Approach

In a real-world implementation of the proposed system, a camera would wirelessly transmit estimated vehicle positions and associated uncertainty to vehicles in its field of view.

To enable position estimation, the camera needs to know its own world position and orientation, as well as the position and orientation of the ground plane. In in this work, I assume the ground plane to be flat (a locally reasonable assumption, especially along roads), and equal to the $Z = 0$ plane. I report the position (x, y) of the vehicle as the estimated geometric center of the vehicle on the ground plane. Note that in practice, using the midpoint of the front or rear axle, rather than the center, may be a more consistent reference between vehicles.

Once the computer vision algorithm running on the camera estimates the center of the vehicle, it must also be associated with an uncertainty. I simplify to a circular error radius, r, which is transmitted as a $2\sigma^2 I$ covariance with the state vector (x, y). The error radius directly determines the operational capability of the autonomous vehicles, thus must be accurately represented.

There are three components that contribute to the error radius:

1. Error in the camera's own location and orientation
2. The algorithmic error in the estimation of the pixel that represents the estimated center of the vehicle
3. The ground area covered by the pixel that represents the estimated center of the vehicle

These will be analysed in turn. Note that I assume calibrated cameras without distortions, and no motion blur.

5.2.2 Location and Orientation Error

Of the three sources of error, this one is the most difficult to reason about. Many transport authorities have standard mounting specifications, including mount heights, for cameras [11]. However, none of these specify the tolerances in location and orientation, and we will have to make some educated guesses.

I assume that the position and orientation error is normally distributed, with 95% of positions within 0.03 meters in any direction, and orientation within a 0.5 degree cone about the principal axis 95% of the time. In mathematical terms, this means Δ_x, Δ_y, and Δ_z are all sampled from $\mathcal{N}(0, (0.03/2)^2)$. The orientation error is given as $\Delta_{orientation} \sim \mathcal{N}(0, (0.5/2)^{\circ}2)$. These tight bounds could be achieved using some sort of post-installation calibration procedure.

To translate these values into an uncertainty on the ground plane, consider that any horizontal camera movement (Δ_x, Δ_y) from its true position simply translates the predicted vehicle location and therefore increases its error by the length of the translation. I call this combined value err_{xy} and at two standard deviations it is bounded above by $|err_{xy}| = 0.03$.

Vertical translation, $\Delta_z \sim \mathcal{N}(0, (0.03/2)^2)$ of the camera induces an error as $err_z(d, h) = \Delta_z \frac{d}{h}$ where d is the ground distance to the vehicle (on the XY plane), and h is the true intended height of the camera.

Thus, the total contribution from the camera's positional error to the error radius will (95% of the time) be bounded by $(err_{xy} + err_z(d, h))$, which evaluates to

$$err_{xyz}(h, d) = 0.03 + \frac{0.03d}{h}$$

Similarly, it can be shown that using a conical error region about the desired principal axis of magnitude $\Delta_{orientation} \sim \mathcal{N}(0, (0.5/2)^\circ)$, for a camera at height h, and ground distance to the vehicle d, the error induced in the estimation of the vehicle's center is

$$err_{orientation}(h, d) = h \tan\bigl(\tan^{-1}(d/h) + \Delta_{orientation}\bigr) - d$$

TODO graphic showing two types of orientation error to help reader visualize what is being calculated

5.2.3 Algorithmic Error

The vehicle localization algorithm needs to scan the pixel plane for vehicles, and return pixel representing the center of vehicles on the ground plane. The center of the pixel projected onto the ground plane is taken as the vehicle center.

This process can be arbitrarily complex. I assume that at the time of deployment, computer vision algorithms are very good at this task, but may still be wrong by several pixels. I analyze the effect of this inaccuracy in the model validation Section 5.3 and refer to the Euclidean distance in the pixel plane from the true center pixel to the calculated one as $\Delta_{alg} \sim \mathcal{N}(0, \eta_{alg}/2)$.

5.2.4 Error from Camera Limitations

Lastly, the error induced by the camera must be accounted for. Consider that in the limit, a vehicle is covered by exactly one pixel, meaning any estimate of its position is at least as uncertain as the ground area covered by that pixel. On the other hand, if a vehicle exactly fills the camera plane and the localization algorithm perfectly determines the center of the vehicle in the pixel plane, the estimate still cannot be better than the area covered by the single pixel.

I call the area covered by a pixel p, $A(p)$. $A(p)$ is also a function of the camera position and orientation, and the camera parameters (field of view and resolution), but these are omitted here as they are constant for any given camera.

Correspondingly, an approximate upper bound on the longest distance within a pixel can be calculated as the diagonal of a square, resulting in a ground distance of $err_{pixel}(p) = \sqrt{(2A(p))}$.

5.2.5 Complete Error Model

I now combine the three sources of error to determine an approximate error radius for the vehicle localization.

$$r(h, d, p) = err_{xyz}(h, d) + err_{orientation}(h, d) + \Delta_{alg} * err_{pixel}(p)$$

For any individual camera, the err_{xyz} and $err_{orientation}$ terms introduce a directed bias for any point in the pixel plane, though over many installed cameras these terms have a normal distribution. The Δ_{alg} is modeled as normal as well. To take advantage of this, I also multiply each with a tuning parameter, related to their variances, which I use to trade off error bound size versus error bound accuracy. The err_{pixel} term is systematic i and a complex function of camera parameters and estimated vehicle location, and the only component with a non-zero mean over a large number of samples.

To validate r in simulation, when a vehicle is visible to a camera, I project the ground truth vehicle center to the (slightly-mispositioned) camera, which automatically incorporates the positional and orientational errors. The algorithmic error shifts the incident pixel a distance sampled according to the distribution of Δ_{alg}, in a randomly sampled direction. Finally, the pixel error is computed from the area of the chosen pixel.

5.3 Error Model Validation

In this section I evaluate the error radius function developed above. It is a good error model if,

1. For a given estimated vehicle location, the true location should be within the predicted error radius most of the time. My formulation aims for 95% accuracy.
2. It is not over-conservative: the tighter the bound, the more useful the update is for vehicle navigation.

I also examine how the different components that make up the error radius vary as the pitch of the camera varies, followed by a brief look at the effect of algorithmic inaccuracy. Experiments were performed with the camera placed at a height of 6 meters above the ground, and a fixed field of view of 60 degrees horizontally and vertically. Rotation about the Z axis was ignored as it does not affect results.

5.3.1 Quality of r as an Error Bound

For this and the following subsection I assume a perfect algorithm, setting Δ_{alg} to 1.

Table 5.1: Success Rate of r as an Error Bound

Total Samples	# Within Error Bound	% Within Error Bound
125000	123865	99.092

Table 5.1 shows that more than 99% of 125000 sampled camera positions and vehicle positions, the distance from the predicted vehicle position to its actual position is within is within the calculated error radius. This is actually better than intended, since the construction was for 95% accuracy.

It is easy to achieve an error bound that is always valid: simply set it very high. Figure 5.2 shows two important characteristics. Firstly, the difference between the predicted error radius and the actual error distance is small, meaning it is not an excessively high bound. Secondly, when the bound is *incorrect* (very small region less than zero in blue) the true error only slightly exceeds the error bound.

To look at how tight the error radius computation is, I present the same chart but plot the residual between predicted radius and actual error distance as a proportion of the predicted radius. Figure 5.3 shows that for the majority of predictions, the error radius is far too large – most commonly it is 3 times too large.

However, consider that each sample is drawn from a near-normal distribution, and the computed bound is in the tail of that distribution – we thus expect that most of the time the bound will be much larger than the actual sample

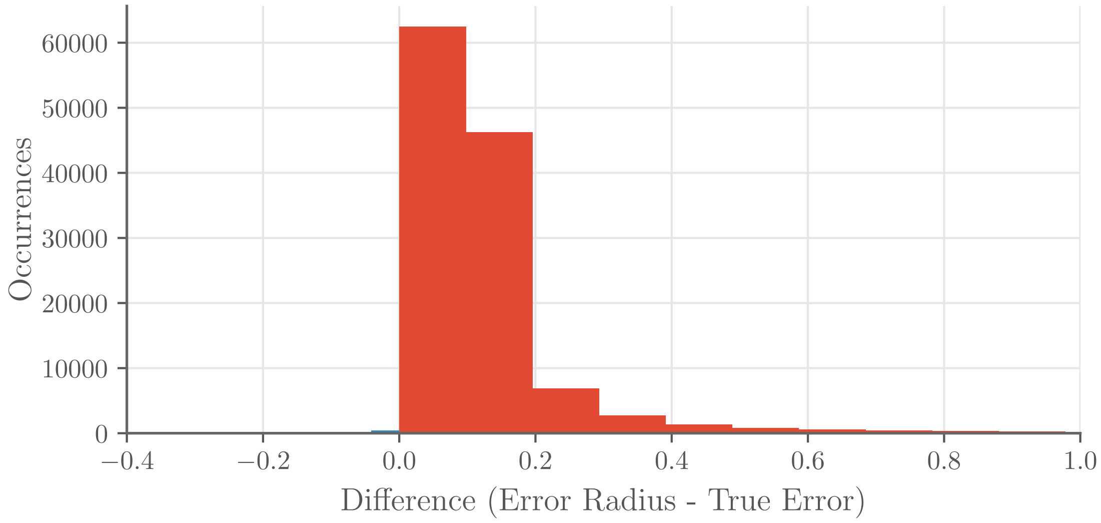

Figure 5.2: Computed radius minus actual error distance.

value. Attempts to make the bound tighter resulted in a loss of the 99% accuracy presented above.

*** TODO - should probably back this up further to make sure it makes sense... ***

These results suggests the r is very good at modeling the true error, even though during construction I did not model interplay between some of the factors (eg. how orientation or positional error affects pixel area inaccuracy).

5.3.2 Error Radius as a function of Pitch Angle

The XY offset of a camera induces a constant distance on the ground plane. However, all other components modeled in the error bound are dependent on pitch angle. Notably, pixel areas diverge as the camera angle approaches horizontal. Figure 5.4 illustrates the components of the error radius as a function of pitch.

This gives us an intuition for how each factor impacts the radius. As expected, orientation and pixel area contributions come to dominate, while at

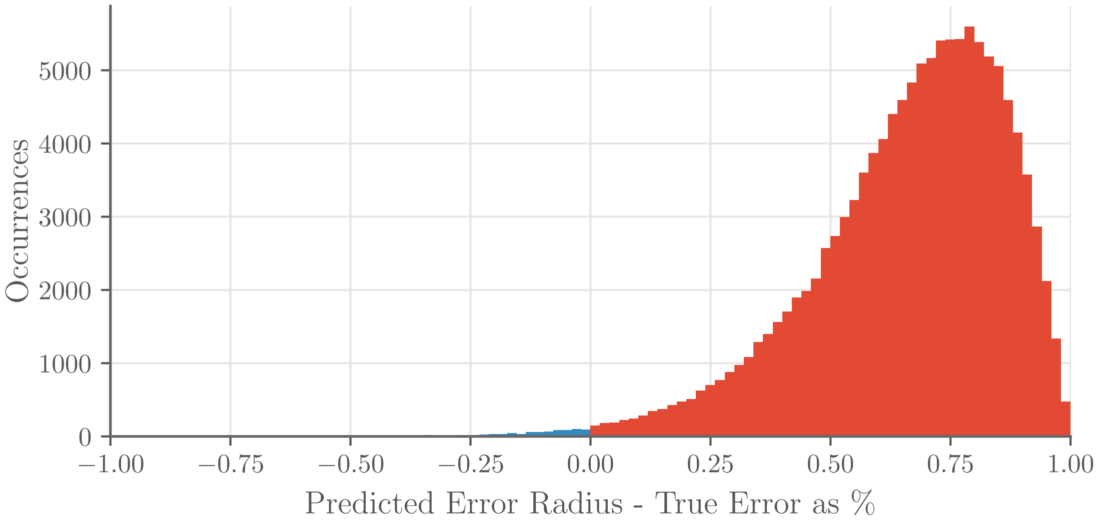

Figure 5.3: Radius error bound minus actual error distance as a proportion of radius.

90°, when the camera is pointing straight down, the dominant error is the camera's horizontal translation.

Also note that the pixel-area component of the error is the only component that can be affected by the camera's configuration and the algorithmic accuracy in detecting the center pixel of the vehicle. This leads to some fundamental limits discussed in Section 5.4.

5.3.3 Impact of Algorithmic Inaccuracy

The prior subsections were all presented without any algorithmic inaccuracies. However, no computer vision algorithm can be expected to perfectly determine the pixel containing the center of the vehicle.

Figure 5.5 shows how the mean error radius changes as a function of the algorithmic error in red. In blue, I show that the accuracy as a bound is minimally influenced by the introduction of the extra error source. This implies that my bound of algorithmic error is indeed well matched to the growth of the actual error.

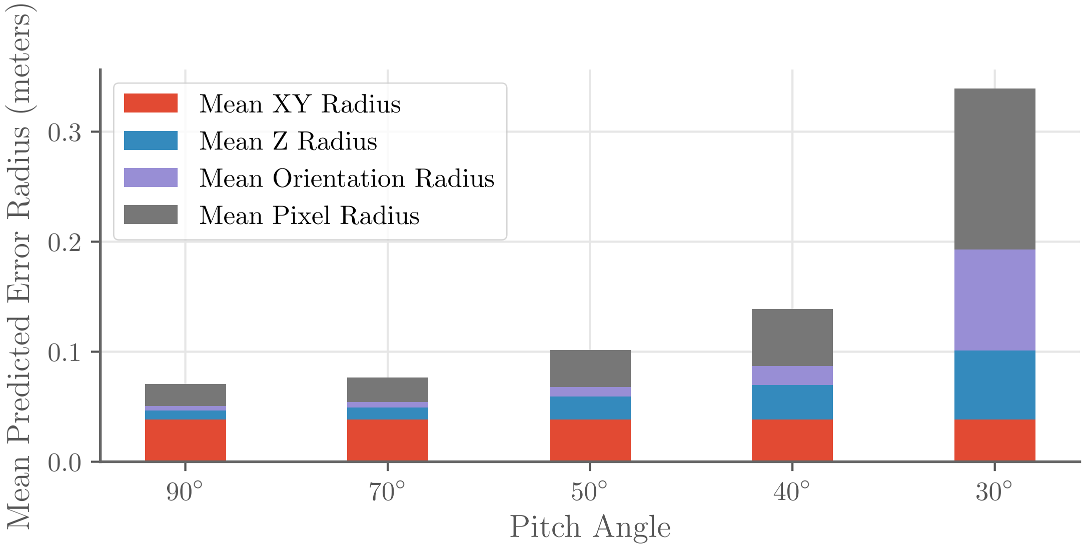

Figure 5.4: How r varies as a function pitch angle.

5.4 Limits and Implications

Having validated that the error model serves as a good bound on the real errors, we can use the function r to reason about fundamental limits of the proposed system.

One interesting questions to ask is: if we had infinite resolution, how well could we perform? Alternatively, how much accuracy do we gain from increasing resolution?

*** TODO update this figure with 2 standard deviations ***

Figure 5.6 shows that the mean error bound stops decreasing significantly beyond a resolution of 2000 by 2000 pixels. A smaller pitch angle gains further from higher resolution, which is in line with what we expect from Figure 5.4 where we saw that more horizontal cameras have a larger part of their error bound stemming from pixel area uncertainty, which can in turn be reduced further by increasing resolution.

A similar behavior can be had by decreasing the field of view, which was for

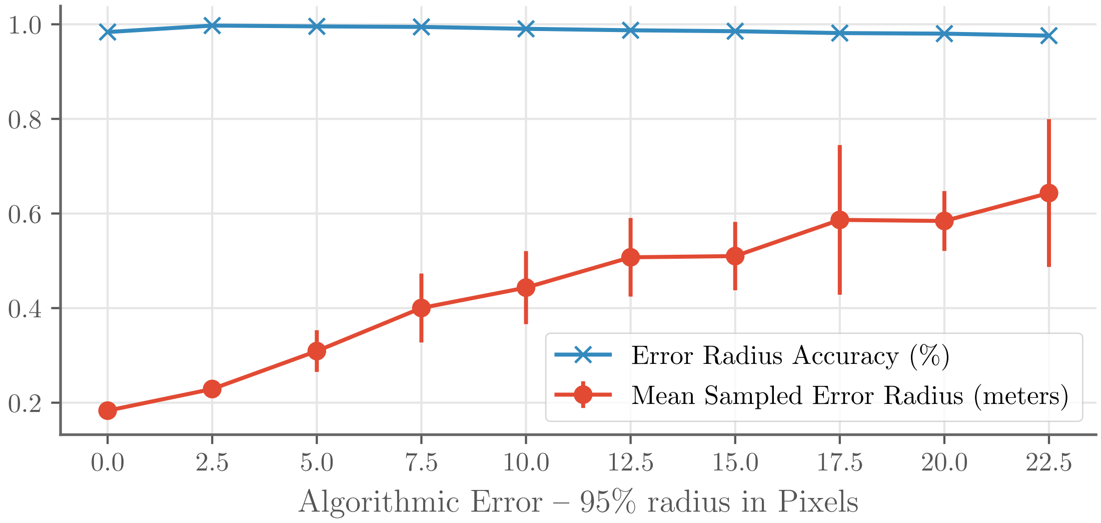

Figure 5.5: The effect on the mean value of r, and bound accuracy, as the algorithmic inaccuracy increases. Error bars indicate two standard deviations.

these experiments set to 60° horizontally and vertically. In the next chapter, I fix the resolution and only optimize for the field of view, since a higher resolution always increases the performance of the proposed system and is thus not interesting.

Figure 5.6 was created assuming the following 95% bounds: positional error within 3 centimeters in any direction, orientational error within a 5° diameter cone about the intended axis, and an algorithmic inaccuracy radius of 4 pixels. Table 5.2 shows the tightest reasonable bounds (sampled from a large number of cameras instantiated according to the allowed variations, and across various world vehicle positions).

Table 5.2: Sampled Prediction Error Bounds using 2000x2000 resolution

Pitch	**Mean (m)**	**Median (m)**	95^{th} **Percentile (m)**
90°	0.1035	0.1035	0.1172
70°	0.1163	0.1075	0.1658
50°	0.1824	0.1454	0.3866
40°	0.3167	0.1884	1.022

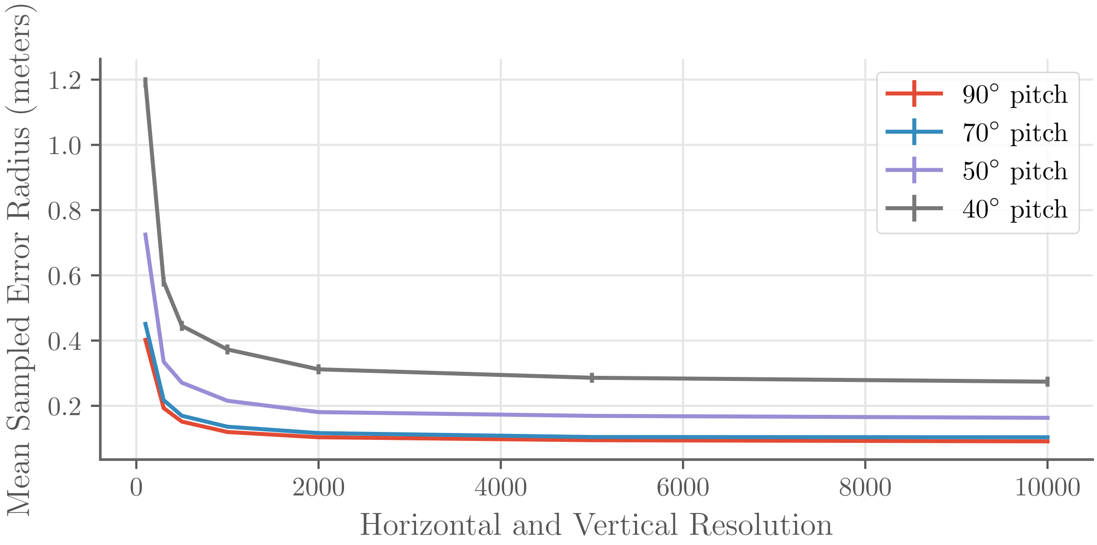

Figure 5.6: How increased resolution can increase the error bound at different pitch angles.

*** TODO update this table with bottom 25% as well, this might be the only region being used as vehicle goes by ***

TODO talk about required accuracy for navigation... generally quoted from 10-25 centimeters so we are just about ok if the pitch isn't too flat... But if the pitch IS low the error bound will still be small toward the bottom of the pixel plane =¿¿¿ discuss general feasibility of cameras and installation constraints!

5.5 Prior Work

*** Is this needed? ***

I relate the derivation above to some of the literature which focuses on landmark placement or camera optimization for surveillance.

Bodor et al. optimize the surveillance of an area by considering line segments, representing piecewise paths, on the ground plane from a fixed height above the ground. The authors consider one variable - pitch, and during the formulation of their objective function consider the number of pixels r filled by the target. This is approximated using the distance to the target and its angle relative to the camera (imagine a car rotating in place, shrinking the number of pixels it takes up)....

5.6 Conclusion

This chapter has developed and evaluated the camera sensor model used in the following chapter to provide correctional updates to an autonomous vehicle. To achieve this, the center of the vehicle needs to be estimated in world coordinates. This is associated with an error, which informs how trustworthy an individual update is.

The error bound developed here, r, is a function of camera height, camera pixel areas, and the horizontal distance to the vehicle center. It was shown to be an effective upper bound, 99% of the time, and small enough to be usable in a real-world implementation (assuming the installation bounds can be achieved). The effect of pitch angle was extensively analyzed, and Section 5.4 explored the impact of resolution in the limit, providing a fundamental limit on the effectiveness of the proposed system, as well as some real-world constraints requirements in terms of installation and algorithmic accuracy.

Chapter 6

Camera Placement

6.1 Single Camera Placement – Localization

6.1.1 A Cheaper Objective Function

Pros/Cons

6.1.2 Straight Roads

6.1.3 Relation to Curvature

6.1.4 Aside: Intersection

*** optional – examine if time remaining at the end ***

Because we can in this case! ... may be a pain to implement in current code of piecewise paths...

6.1.5 Varying Pitch, Yaw, and Horizontal and Vertical FoV

* hill climbing (put in background?)

6.2 Single Camera Placement – Navigation

6.2.1 Full Objection Function

* Monte Carlo approximation of entropy

* may need to outline maximizing mutual information =¿

* to minimizing entropy. Relate to Beinhofer's evaluation

* TODO Try out just using trace of covariance matrix...

* optional: research other localization/navigation objectives used again, related log(det) to tr(cov) to error radius to entropy...

6.2.2 Straight Roads

6.2.3 Relation to Curvature

6.2.4 Comparison to Simpler Objection Function

6.3 Multiple Camera Placement

* Discuss submodularity

* test submodularity by sampling – make sure minimum overlap of camera ground areas!

* straight roads + 1 curvature

* propose cascaded optimization – choose possible sensor locations, use cheaper objective to find top 1 or 2 positions or orientations, add these to the set of possible locations and orientations

6.4 Conclusion

Chapter 7

Optimizing Road Construction for Navigation

* brief - look at how to connect two points if various obstacles are put in the way * maybe try a triangular point an and rectangular block

Chapter 8

Summary and Conclusion

Bibliography

[1] Bruno M Scherzinger. “Precise robust positioning with inertial/GPS RTK”. In: *Proceedings of the 13th International Technical Meeting fo the Satellite Division of the Institute of Navigation (ION GPS)*. 2000, pp. 115–162.

[2] Juho Kannala and Sami S Brandt. “A generic camera model and calibration method for conventional, wide-angle, and fish-eye lenses”. In: *IEEE transactions on pattern analysis and machine intelligence* 28.8 (2006), pp. 1335–1340.

[3] Robert Bodor et al. “Optimal camera placement for automated surveillance tasks”. In: *Journal of Intelligent and Robotic Systems* 50.3 (2007), pp. 257–295.

[4] Norman Mattern, Robin Schubert, and Gerd Wanielik. “High-accurate vehicle localization using digital maps and coherency images”. In: *Intelligent Vehicles Symposium (IV), 2010 IEEE*. IEEE. 2010, pp. 462–469.

[5] Thomas K Sharpless, Bruno Postle, and Daniel M German. “Pannini: a new projection for rendering wide angle perspective images”. In: *Proceedings of the Sixth international conference on Computational Aesthetics in Graphics, Visualization and Imaging*. Eurographics Association. 2010, pp. 9–16.

[6] Alberto Hata and Denis Wolf. “Road marking detection using LIDAR reflective intensity data and its application to vehicle localization”. In: *Intelligent Transportation Systems (ITSC), 2014 IEEE 17th International Conference on*. IEEE. 2014, pp. 584–589.

[7] Julius Ziegler et al. “Video based localization for bertha”. In: *Intelligent Vehicles Symposium Proceedings, 2014 IEEE*. IEEE. 2014, pp. 1231–1238.

[8] Shunsuke Miura et al. “GPS error correction with pseudorange evaluation using three-dimensional maps”. In: *IEEE Transactions on Intelligent Transportation Systems* 16.6 (2015), pp. 3104–3115.

[9] US Air Force. *GPS Accuracy.* 2016. URL: https://www.gps.gov/systems/gps/performance/accuracy/ (visited on 05/19/2018).

[10] *Selecting the Right Camera for Intelligent Traffic Systems.* Lumenera. 2017.

[11] Transport for London. *Streetscape Guideance.* 2017.

[12] Rafael Vivacqua, Raquel Vassallo, and Felipe Martins. “A Low Cost Sensors Approach for Accurate Vehicle Localization and Autonomous Driving Application”. In: *Sensors* 17.10 (2017), p. 2359.

[13] Shih-Chieh Lin et al. “The Architectural Implications of Autonomous Driving: Constraints and Acceleration”. In: *Proceedings of the Twenty-Third International Conference on Architectural Support for Programming Languages and Operating Systems.* ACM. 2018, pp. 751–766.

[14] Margaret M Fleck. “Perspective Projection: the Wrong Imaging Model. 1995”. In: *Google Scholar* ().