

SURF report - Road following based on video and LADAR data

Gustav Lindström

1 Introduction

The possibility of creating a system that acts like a human being has long been a goal of researchers. Just a few years ago scientists managed to create a computer capable of playing chess better than any human being and now the next step is to make a system that can perform human tasks in a less controlled environment. These kinds of robotic systems will eventually revolutionize the way we live, taking care of repetitious and tedious tasks such as cleaning, cooking and even driving the car.

The DARPA Grand Challenge(DGC) is a step in this direction, and consists of creating a car that can autonomously drive 175 miles through the Nevada desert in less than 10 hours. The challenge, sponsored by DARPA (defense advanced research project agency), had its first competition last year (2004), resulting in the best vehicle finishing just over 7 miles of the 140 mile course. However, during the time from last year, several new teams have entered the challenge and much needed scientific progress has been made.

2 Alice

The Caltech contribution to DGC 2005, Alice, is a project driven mainly by undergraduates from Caltech and SURF students from around the world. The vehicle is a off-road adapted Ford E-350 with several additions in the form of actuators for steering and throttle control, as well sensing capabilities. The sensing consists of 5 Ladars (Laser Detection and Ranging), and 5 black and white cameras, one of which is used for road following.

From all the sensing data a height map is created, and using methods similar to image processing methods, this data is converted to allowable speeds. If, for example, a certain segment of the height map seems to be bumpy, the corresponding allowable speed is lowered. Finally, this speed map is sent to a module responsible for planning an optimal route through the terrain.

3 Road following

The vehicle would be fully operational without knowledge about roads and other man-made objects in its vicinity; all that is needed is detection of height

differences in its vicinity. The information added by a road detection system however lets the vehicle travel faster and in a more human fashion.

For example, consider a human driver traveling in an off-road environment. While driving on a road a human makes the implicit assumption that the surface is relatively flat and that the road does not end abruptly on the other side of a hill. Using the same assumptions in an autonomous vehicle makes it possible to drive significantly faster and to avoid getting into dead ends.

3.1 Problem formulation

A general road following algorithm developed and implemented by Christopher Rasmussen[1] was used on Alice. The algorithm takes image data from a centrally mounted camera on the vehicle, calculates the dominant directions of the textures in the image and, from this, derives the vanishing point of the road. In combination with this data, distance measurements from the Ladars were used to calculate the width of the road and the lateral offset of the road compared to the vehicle.

All this processing results in a number of control points, relative to the vehicle, that specify the position and the width of the road ahead. The goal of this report is to explain how this information was integrated into the overall system and how the vehicle was made to react correctly on road information.

3.2 Road painting

There are two main ways of integrating road data into the overall architecture. The first one tested consists of sending the detected road points straight to the vehicle trajectory system, thus making the vehicle drive straight down along the center of the road. The problem with this approach however, is if there is a big obstacle in the middle of the road. Obviously there are ways to abort the road following if obstacles are detected etc, however other scenarios can be constructed in which the complexity of this approach grows.

Instead the method used in the final race configuration consists of painting the road into the map as a speed bonus. This makes use of the underlying framework with a speed map generator and a planner, and just merges the new information into the existing system, thus avoiding the need for special case treatment of obstacles etc.

3.2.1 Road rasterization

The data supplied to the road painting algorithm was a series of points and corresponding road widths in those points; this data was then transformed to a polygon representing the road. As the road width varied, so did the width of the polygon, the problem however was to determine in which direction to measure the width of the road. The simplest solution, used in the painting of the vehicles corridor, is to use circles centered in the road points with the diameter

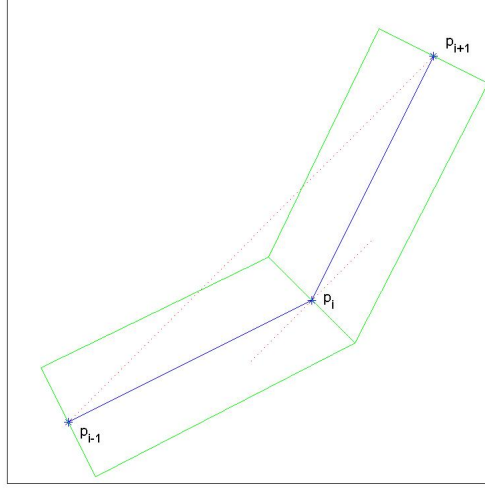


Fig. 1: Polygon generation from road control points

corresponding to the width of the road. The problem with this approach is the computational expense of rasterizing circles.

Therefore a different approach was followed. If we consider the direction of the road in a point p_i to be that of the directional vector between p_{i-1} and p_{i+1} , it is easy to extend the width of the road perpendicular to this vector and then join these edge points to create the road polygon, see figure 1.

As for the rendering part of the rasterization, the polygon was splitted into convex sections by only considering the 4 points from p_i and p_{i+1} in each iteration. Then for each convex section a standard scan-line-conversion algorithm was used to render it into the map.

3.2.2 Uniform vs sloped road

The next problem was to determine how the speed bonus was to be painted. During the first trials the road was painted with a uniform speed bonus, this however made it unclear for the player where to place the vehicle lateral within the road, resulting in driving performance very close to the roads edge. To improve upon this, the algorithm was altered such that the speed bonus was proportional to the distance (actually the square of the distance for computational optimality) from the road center. This is easily achieved by using some vector algebra.

Let p_0 and p_1 be the start respectively end points of the road segment, and let q be the point currently painted. The square of the distance is then simply

given by:

$$d^2 = \left| (q - p_0) - \frac{(p_1 - p_0) \cdot (q - p_0)}{|p_1 - p_0|^2} (p_1 - p_0) \right|^2$$

This algorithm worked fine in most situations, however when the width of the road varied a lot, some problems occurred. As only the distance from the road's centerline is considered, all information about the actual width of the road is discarded. On a wide road, it does not matter very much if the vehicle is off by 1m from the center line of the road, thus the speed bonus should still be large. However on a narrow road, 1m off to the side could mean that we are driving dangerously close to the edge of the road. The next improvement of the algorithm was therefore to paint the speed bonus in proportion to the distance from the center line relatively to the width of the road. In the same way as above, the equation for this can be derived straight from linear algebra.

Let as before p_0 and p_1 be the start respectively end points of the road segment, and let q be the point currently painted. Let also r_0 and r_1 represent the half width (radius) of the road in point p_0 and p_1 respectively. As we already have an equation for the distance from the center line, we need to determine the width of the road in the specific point. For this we assume the boundary of the road is a straight line and therefore the width of the road is proportional to how far along the road segment we are. The distance along the road segment can be calculated as:

$$t = \frac{(p_1 - p_0) \cdot (q - p_0)}{|p_1 - p_0|^2}$$

Thus the radius at q is:

$$r = r_0 + t(r_1 - r_0)$$

And the relative distance, rd , is calculated as:

$$rd^2 = \frac{d^2}{r^2}$$

All these computations were further optimized such that each point rendered only required 8 multiplications and 1 division.

3.3 No road data

Once the road was painting correctly and the vehicle was able to track the road through the information painted into the map, a new problem arose. What happens if the vehicle goes off-road, or the road turns too quickly for the following algorithm to keep track of it? The first part, implemented by Christopher Rasmussen, consisted of calculating the Kullback-Leibler divergence between the dominant orientation distribution in the image, and the expected random distribution. Using this measurement, a number of thresholds were derived from test data and the algorithm was able to detect the no-road situation.

The second issue, with rapidly turning roads, was initially handled with the above approach; however the inertia in the algorithm did not always allow the

road painting to be turned off until it was too late. Therefore the information in the RDDF was used instead. Under the assumption that, when driving on dirt roads, the RDDF center line somewhat follows the road curvature, said curvature was calculated over the following 30m and if the curvature was greater than the maximal road curvature detectable, the road painting was instantaneously turned off. The result of this being that the car slowed down additionally when approaching a steep curve.

4 Results

During testing, impressive following was achieved using the road following algorithm, with stretches of up to 15 miles of dirt roads followed completely autonomously. The exact performance of the road following algorithm is very hard to estimate as it is a part of a larger system, and not the only thing giving commands to the vehicle. From inspection of test logs and during tests, the algorithm seemed to perform very well during daylight conditions on normal dirt roads. The main problems that were uncovered were during sunrise/sunset when shadows, especially from the vehicle itself, produced spurious dominant orientations and at one occasion even sent the vehicle running off into the side of the road.

The other problem uncovered was that of driving on big open areas. When no clear road was present, but there was still texture on the ground, the dominant orientations picked up by the algorithm were not always consistent, sending the vehicle swirling over the plains.

5 Further improvements

The main disadvantage of the road following algorithm is its inability to track a curving road, although the vanishing point gives a good general idea of the road, it is not enough when driving on winding roads. Another related problem is that of tracking merging and splitting roads, when the road ahead splits up into two roads, the algorithm currently chooses one of them on random. An alternative here would be to either track both of the road segments, or to use information present in the RDDF to decide on which segment to track.

6 Conclusion

Although the performance of Alice on race day, 21st of 23 teams, was not as good as we had hoped for, it was interesting to work on the project. Although the project was a research project, much of the work was not pure research. It felt like the project more resembled a real life engineering project in that respect, with a solid deadline and a hunt for solutions that worked well enough and that could be implemented fast and reliably. It has therefore been a great experience which extends beyond what you normally learn in the classroom, and has given a taste of real life engineering projects.

Apart from this, the main lesson learned on the Caltech DGC 2005 team is the importance of clear leadership. Many of the decisions needed to be taken, where not taken by the people responsible or at the time when they where needed. Much unnecessary work was performed because the decisions on canceling projects were delayed, as nobody wanted to be the one terminating a projekt. Further the testing process was not as methodical as would have been needed. Instead of testing every component separately in simulation and fixing every little unexpected behavior as it came up, the testing was more of a "happy hacking". The vehicle was driven 2h out into the desert, everything was started up just to discover that something was not working. Several 100s of man hours of work where probably lost due to people having to sit around in the desert just waiting for one module to get fixed. Once the vehicle was running, unexpected errors where often treated with a "lets restart" comment, without noting the exact reason for the problem or even trying to actively isolate the error. All this resulted in a system that felt very unstable and unreliable. For example the comment when starting the Ladar drivers that 3-4 initial errors were to be expected (if starting at all), is not what you expect from a system suppose to run for 10h without human intervention.

References

- 1 . Grouping Dominant Orientations for Ill-Structured Road Following, Christopher Rasmussen, Computer Vision and Pattern Recognition, 2004