2008 IEEE International Conference on
Robotics and Automation
Pasadena, CA, USA, May 19-23, 2008

Image-Based Path Planning for Outdoor Mobile Robots

Mark Ollis, Wesley H. Huang, Michael Happold, and Brian A. Stancil

Abstract— **Traditionally, path planning for field robotic systems is performed in Cartesian space: sensor readings are transformed into terrain costs in a (Cartesian) costmap, and a path to the goal is planned in that map. In this paper, we propose a new approach: planning a path for the robot in the image-space of an on-board camera. We apply a learned color-to-cost mapping to transform a raw image into a cost-image, which then undergoes a pseudo-configuration-space transform. We search in the resulting cost-image for a path to the projected goal point in the image. One benefit of our approach is the ability to react to obstacles at ranges well beyond our 3D sensor range — independent testing has confirmed our system has effectively reacted to obstacles at a range of 93 m while our stereo sensor provides reliable data only up to 5 m away. We describe the details of our technique and the results from testing under the DARPA LAGR and UPI programs.**

I. Introduction

Autonomous robots operating in outdoor environments have traditionally done path planning in Cartesian maps. There are good reasons for this: it is simplest to fuse data from multiple sensors in a Cartesian map, and it is easiest to plan a shortest path to the goal in a Cartesian map.

However, the resulting paths are good only to the extent that the Cartesian map contains an accurate representation of the world. Field robotic systems using Cartesian maps have two weaknesses in this respect.

The first weakness is that obstacles can be aliased when transformed from the sensor's space to Cartesian space: solid regions may project with gaps, or thin corridors may vanish completely. For example, Figure 2 shows a scene in which a continuous "wall" of vegetation becomes discontinuous when projected into a Cartesian map.

The second weakness is the limited range of common commercially available range sensors. The SICK LADAR units used on many robots provide reliable obstacle information only up to approximately 25 m. (We consider obstacle detection to be reliable when it provides stable readings with sufficient range and size accuracy to commit the obstacle to a Cartesian map.) The PointGray Bumblebee stereo units we are using provide reliable obstacle detection to a range of approximately 5 m. Without long-distance perception, robots in natural outdoor terrain tend to be myopic, which leads to inefficient navigation.

We propose to address both of these weaknesses by planning a path for the robot in the image-space of an on-board camera. Our approach is as follows:

This work was done at Applied Perception, Inc., 220 Executive Drive, Suite 400, Cranberry Township, PA 16066, USA. Wes Huang and Brian Stancil are with Applied Perception, Mark Ollis is now with Lockheed Martin, and Michael Happold is now with Carnegie Mellon University. The corresponding author is Wes Huang, wes@appliedperception.com.

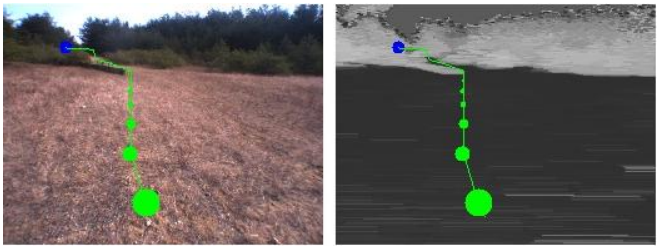

Fig. 1. Image from the DARPA LAGR Test 7 course (left) and the result from our image-based planner (right).

Fig. 2. A scene containing a "wall" of vegetation (left) that gets projected into the Cartesian cost map (right) as a discontinuous row of obstacles (red pixels). The robot is located at the end of the green path facing the lower right corner of the cost map.

- Take a raw image from the camera and convert it into a cost-image, where a pixel value represents the terrain cost of the patch of the world projected onto that pixel.
- Apply a pseudo-configuration-space transform to the cost image to account for the size of the robot.
- Project the goal point into the image-space (unless the goal can be visually identified in the image).
- Plan a pixel-to-pixel path from a pixel at the bottom of the image (a point right in front of the robot) to the goal pixel

By planning directly in the image-space, we avoid the aliasing issues from projecting obstacles into a Cartesian map. By evaluating terrain cost directly in the image, we can recognize and react to obstacles far beyond the maximum range of our range sensors.

As an example, consider the scene shown in Figure 1. A short fence creates part of a cul-de-sac that lies on the straight-line path to the goal. Without long-distance perception, the robot would drive straight towards the goal, entering the cul-de-sac. (This cul-de-sac is deep enough that our stereo vision cannot detect the end until it drives several meters into the cul-de-sac.) However, from the robot's current position, it is clear from the image that the grassy field extends around to the right, beyond the thicket of shrubs

at the end of the cul-de-sac.

There are some drawbacks to image-based planning which make it unsuitable as the sole path planner for a robot:

- The goal point may not project into the image — while we can plan a path to the nearest pixel on the boundary of the image, this is only useful if the goal is not too far beyond the edge of the image.
- The image-based planner will not always return a good path — if there is not a reasonably clear path most of the way to the goal, the resulting path is generally not useful. (Fortunately, we can use the cost of the path to detect this condition.)
- The resulting path may not be entirely drivable due to the lack of three-dimensional information.

Consequently, we use the image-based planner as a secondary planner in our system. When the image-based planner finds a good path, it can supersede the normal Cartesian planner operation. We pick a point up to 7 m from the robot along the image-based path and use it as a subgoal for the Cartesian planner, but one could also simply choose to drive the first several meters of the image-based path.

We have found our image-based planner to be a valuable addition to our system. Independent testing through the DARPA LAGR program[1] has shown that our system can perceive and react to obstacles at a range of 93 m.

The rest of the paper is organized as follows. Section II describes the details of our approach, Section III relates the results of independent testing of our implementation, and Section IV examines some of the limitations and properties of image-based planning.

A. Related work

Many researchers have used visual images to do some sort of path planning or navigation without Cartesian maps.

In the area of visual servoing, image-based planning can be used to servo an "eye-in-hand" system: features from the manipulator's current view are compared with the corresponding features in a goal image, and a path for those features is computed in the image space. Zhang and Ostrowski [2] have adapted this approach to navigation for a mobile robot by posing the problem as an optimal control problem that is solved numerically. Rivlin, Shimshoni, and Smolyar [3] take a slightly different approach: they repeatedly estimate the epipolar geometry between the robot's current image and a target image and use the estimated translation direction and rotation to drive the robot.

A related approach to Cartesian path planning has been demonstrated by Howard *et al.* [4]. Their path planner operates on a "polar perspective map" which represents the world using a grid in polar coordinates centered on the robot's location with grid cells that become increasingly larger along the radial axis. This polar representation is similar to image-based planning in that terrain further from the robot is represented at a coarser resolution than closer terrain.

[1]For an overview of the LAGR program, see Jackel *et al.* [1].

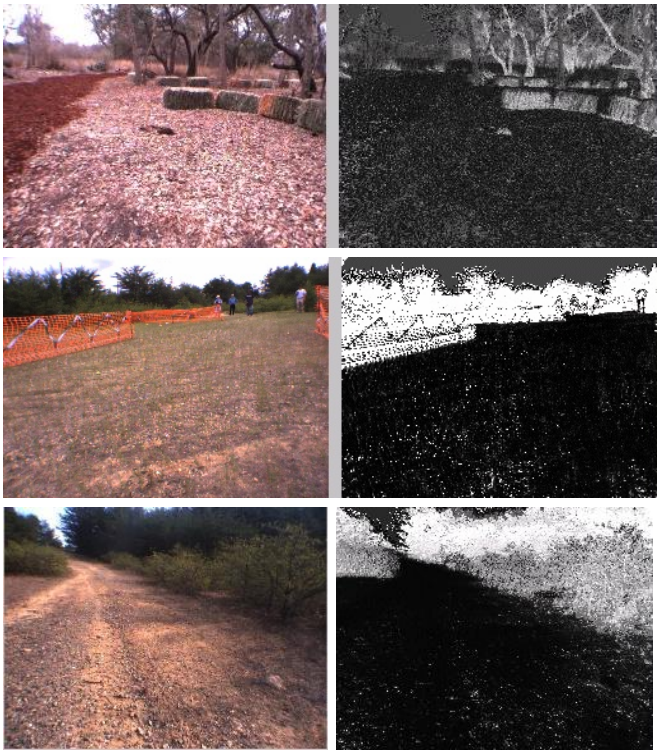

Fig. 3. Example images from the on-board camera (left) and the resulting cost-images (right) using our learned color-to-cost mapping.

II. Approach

A. Cost-image generation

There are a variety of methods that could be applied to generate a cost-image from a single monocular image from an on-board camera: texture classification, object/terrain recognition, image segmentation, etc. The key idea is to accurately label pixels or regions in the image with terrain cost.

We perform this conversion using a learned color-to-cost mapping algorithm [5], [6] that we developed early in the LAGR program. The basic idea behind this technique is that the association between colors and terrain cost in the near-range are an excellent predictor of terrain costs in the distance, at least over short time periods.

We use a neural net classifier to take stereo data (within 5 m of the robot) and determine terrain costs. Our learning element uses the nearby terrain (costs and colors) to produce a color-to-cost mapping which we can then apply to a monocular image to predict terrain costs for terrain far beyond the range of useful stereo data. This mapping can be learned in as few as five or ten seconds, which allows the system to rapidly adapt to new terrain or lighting conditions.

As a simple example, the system might learn that one particular shade of green corresponds to traversable grass while another corresponds to a nontraversable bush. Changes in environment and lighting condition may later make this association misleading or useless, but we have found that such associations are impressively accurate over the scale of a few minutes or a few hundred meters in a variety of

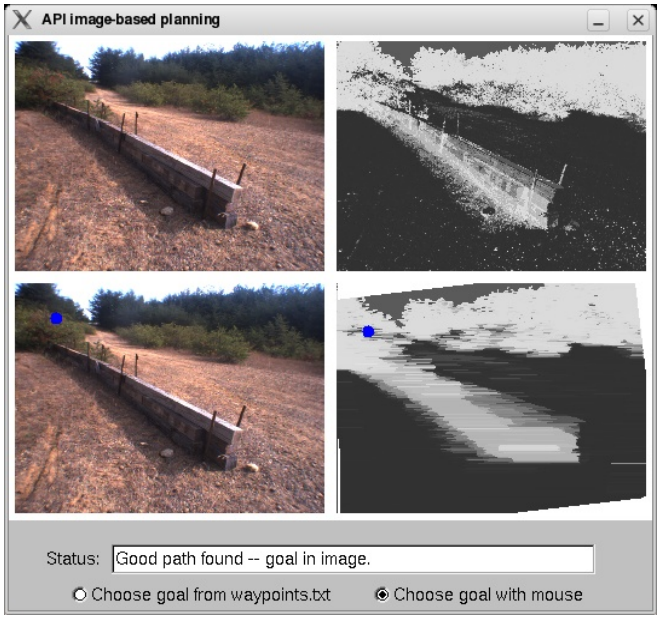

Fig. 4. A raw image (left images), the resulting cost-image (upper right), and the pseudo-configuration-space transformed cost-image (lower right). Note that the transformed cost-image has been rotated to compensate for camera roll.

different environments. Figure 3 shows several examples of our learned color-to-cost mapping.

B. Pseudo-configuration-space transform

As in a traditional Cartesian-space path planner, it is useful to expand the cost-image to compensate for the size of the robot so that we can treat the robot as a point (or single pixel) when planning a path in the image.

We cannot perform a true configuration space transform since we are missing the three-dimensional information required to know the cost of the terrain the robot would actually cover when placed at a given point in the image. However, the image-based planner should only return a path if it finds a low-cost path, so we are most interested in paths that make a lot of progress towards the goal over low-cost terrain. These paths will generally drive monotonically towards the goal, curving only to drive around obstacles.

Therefore, the "pseudo-configuration-space transform" we use will expand terrain costs only horizontally. Our transform will replace the terrain cost of each pixel with the maximum terrain cost within a half robot width to the left or right in that row. (We assume any camera roll has been corrected by rotating the image, so that rows in the image are parallel to the horizon.)

Of course, in the image-space, the robot width is not constant: it is much larger in the foreground than it is in the background. We use a simple approximation to the ground plane to determine the appropriate expansion width for each row. However, we could have used the available stereo information to determine this width. Fortunately, only a very rough estimate of range is needed, especially for obstacles that are far away. As the distance increases to infinity, in fact, the robot width becomes negligibly small in image-space and can be safely ignored.

Figure 4 shows an example of the pseudo-configuration-space transform. Note that obstacles near the bottom of the image are in fact expanded more than obstacles near the top of the image.

C. Goal-point projection

Since we are planning in image-space, we must somehow identify the goal pixel in the image. If the goal is identified with a visual marker, this problem might be easily solved in image-space as well. For our system, however, the goal is specified by 3D GPS coordinates.

Since our vehicle is equipped with enough sensors to obtain reasonably accurate 6-DOF vehicle pose, the goal can be projected into the image without making assumptions about the shape of the actual terrain in front of it (i.e., without fitting a ground plane to the nearby terrain). However, errors in the vehicle pose will cause errors in the projected image-space goal; we discuss this effect in Section IV-A.

Of course, the goal may not project to a pixel in the image at all. We have found, however, that our system can produce useful paths when the goal is off the edge of the image by up to 5–15 degrees. For these cases, we take the goal pixel to be the pixel on the boundary of the image closest to the projected goal point in image-space.

D. Path planning

Planning the path is relatively straightforward: we perform an A* search on the transformed cost-image to find a pixel-to-pixel path to goal pixel. There are a number of details, however, worth mentioning.

We identify a start pixel by taking a point in front of the robot and projecting it into the image-space. Since our camera is fixed with respect to the robot, the start pixel is always the same, unlike the goal pixel which is generally different for each run of the image-based planner. We assume that the variation in terrain right in front of the robot (where the start point is located) is negligible for purposes of identifying the start pixel.

The cost of a move to a pixel (in the transformed cost-image) is computed based on its terrain cost. For our implementation, terrain costs are represented with 8-bit unsigned values: 0 to 89 are all considered (equally) low-cost terrain, and values 90 to 255 are represent a continuum of moderate to high cost. The cost used for the A* search is a function of the (raw) terrain cost c:

$$\text{cost}(c) = \begin{cases} 0.2 & \text{if } c < 90 \\ 0.4\frac{c^4}{90^4} & \text{otherwise} \end{cases} \quad (1)$$

Though somewhat ad-hoc, this is a simple approximation to the cost function used by the original Cartesian planner supplied with the robot; the terrain costs returned by our perception system are attuned to this cost function.

The heuristic function for A* simply takes the straight line distance (in pixels) multiplied by 0.2. Since 0.2 is the minimum cost for a pixel-to-pixel move, this heuristic

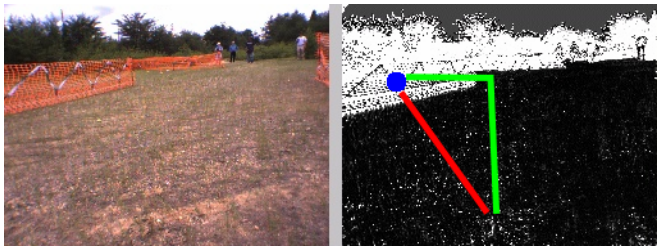

Fig. 5. Illustration of crossing a horizontal edge in a cost-image (red path) that runs into an obstacle and crossing a vertical edge (green path) that goes behind an obstacle to the goal (blue).

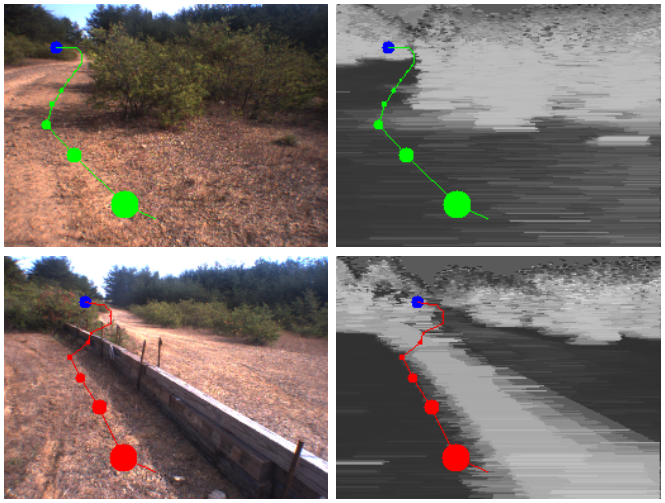

Fig. 6. Example of a low-cost path (green) in the top images and a high-cost path (red) in the bottom images. Note that although paths in the image are planned pixel-to-pixel, we have drawn them using straight lines between sampled points (large dots) from the image-based path for greater clarity.

is admissible, so A* will find the optimal path on the transformed image.

We restrict the directions for moving from pixel to pixel: only horizontal and upwards moves (including the two upwards diagonal moves) are permitted. This is a useful heuristic to focus the search on paths that make monotonic progress towards the goal, and it also decreases the run-time of the planner; however, an equally valid design choice might be allow any 8-connected move in the transformed cost-image.

Another useful heuristic we have added to the image-based planner is to reduce the cost of horizontal moves in the goal row. The cost of these moves is:

$$\text{cost}_{\text{goal-row}}(c) = \min\{\text{cost}(c),\ 0.4\} \tag{2}$$

The design of this heuristic was to encourage paths to move upwards in the image first and then moving horizontally towards the goal. Our intuition behind this is that crossing a vertical edge from low- to high-cost corresponds to a path that goes behind an obstacle, whereas a horizontal low- to high-cost edge tends to be a path that runs into an obstacle on the ground. This heuristic works best when the goal is occluded by a high-cost obstacle. Figure 5 shows an example illustration.

E. Using image-based planned paths

As mentioned earlier, the image-based planner is a secondary path planner in our system. Our scheme for using image-based paths was carefully designed using the following principles:

- Image-based paths provide valuable advice, but they may not be driveable, so we will rely on the Cartesian planner (and our stereo-based local perception) to produce safe paths for the robot to drive.
- Image-based paths provide the most information in the near-range, i.e., which direction to drive to avoid obstacles in the distance, so an image-based path should only be used for a short period.
- For a good image-based path, we should be able to produce a new image-based path from a point further along the path.[2]

Details of our approach are described below.

First of all, the Cartesian planner will only accept low-cost paths from the image-based planner. This is judged using a threshold on the total cost of the image-based path. Figure 6 shows examples of a low-cost and a high-cost path.

The Cartesian planner takes an accepted path (which has been projected onto a ground plane calculated from local stereo data) and finds a point that is either one-third of the way to the goal or 7 m from the robot, whichever is closer. This is used as a temporary goal by the Cartesian planner.

This image-based planning goal will be active for up to 10 seconds or until the robot makes 4 meters of progress towards the real goal or towards the image-based planning goal. If either of these limits is exceeded, planner will revert to the real goal. Usually, a new image-based plan will be accepted before this happens, causing a new image-based planning goal to be used, and the limits to be reset. In order to avoid cyclic behavior,[3] however, we require that the robot make at least 1 meter of progress towards the real goal before another image-based plan is accepted (even if the Cartesian planner has reverted to the real goal in the meantime). Finally, no image-based plan will be accepted if the robot is within 10 m of the (real) goal.

The typical behavior we observe is that the image-based planning goal appears to repeatedly jump ahead, as one image-based planning goal is replaced by another, while the robot keeps making progress toward the goal.

III. Results

A. Results from LAGR

Our system has been independently tested through the DARPA LAGR program. Figure 7 shows an example from Test 22 in which our image-based planner returns a path that avoids an obstacle (several 4 by 8 foot sheets of plywood painted blue) from a range of 93 m.

[2]See Section IV-B for discussion of the effects of this principle.

[3]For example, following an image-based path into a cul-de-sac, leaving it after reverting to the real goal, and then accepting a similar image-based path leading back into the cul-de-sac.

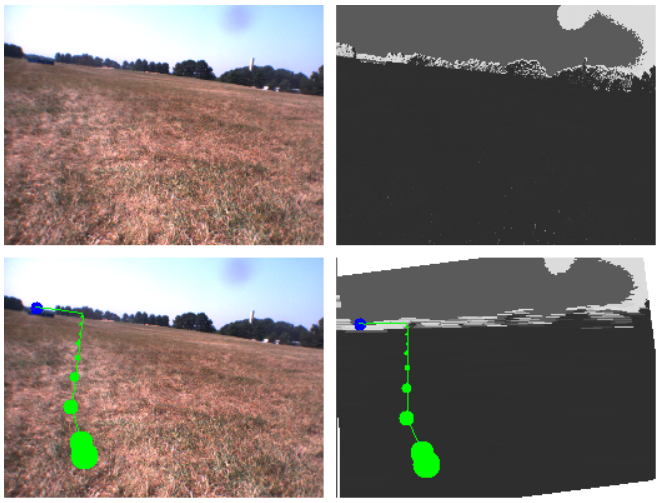

Fig. 7. Example from LAGR Test 22 where the path avoids an obstacle at a distance of 93 m.

Fig. 8. An example of an image-based plan on a dataset from the DARPA UPI program's Crusher vehicle.

We have also ported this algorithm to the DARPA UPI program which uses the Crusher vehicle. Figure 8 shows an image-based plan on that platform.

B. Video attachment

The video attachment to this paper contains three short segments.

- The first segment shows how our color-to-cost mapping is learned in real-time from local stereo data.
- The last two segments show our image-based planner running on log files from LAGR tests. Please note that the robot was not influenced by our image-based planner since it was not running on the robot at the time. The raw image is shown in the left column, the cost-image in the upper right, and the transformed cost-image in the lower left. The image-based path is shown in the bottom row: green paths are low-cost, yellow are moderate-cost, and red are high-cost.

IV. Discussion

A. Accuracy of goal point projection

One of the most common failure modes for image-based planning is the difficulty of accurately identifying the goal pixel in image-space. As mentioned earlier, if the vehicle's 6D pose and the goal point's 3D position are both known precisely, the goal can be projected into the image without error. However, small errors in vehicle pose can result in mis-projections of the goal point, which can have dramatic effects on the image-based plan.

Figure 9 shows an example from a test course for the DARPA LAGR program. The left image correctly shows the goal point behind the trees, and the image-based planner plans around the trees. In the right image, a small error in vehicle pitch cause the goal to appear in front of the trees, resulting in an image-based plan that leads straight ahead.

One related complication is that the elevation values from GPS are not as accurate as the latitude and longitude values,

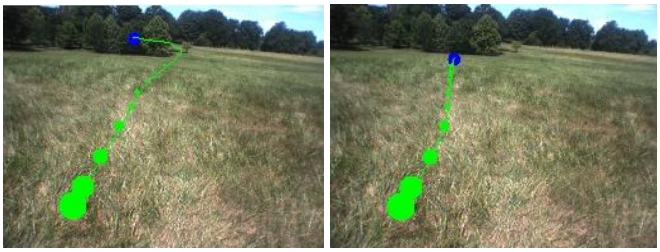

Fig. 9. Example of the effect of pitch error on goal point projection and the image-based planned path.

and elevation errors also have a large effect on projected goal point position. One attempt we have made to solve this problem is to assume that the goal point is at the same elevation as the robot. This has worked well in our tests so far but is an imperfect solution.

B. System behavior

We do observe some oscillation in the robot's path. An image-based path may cause the robot to drive at an angle to the goal. This image-based path eventually expires, and if the goal is too far off the edge of the image, no new image-based plan will be produced. This causes the robot to turn back towards the real goal, at which point a new image-based plan may be produced (causing the robot to turn away from the goal again).

Although not entirely desirable, this oscillation has seemed an acceptable price to pay for the benefits of image-based planning. Allowing the goal to lie off the edge of the image (up to a certain limit) ameliorates this behavior. However, a limit that is too high can cause the robot to take a wide excursion on its way to the goal. Ideally, a camera with a wider (or omnidirectional) field of view would be used to eliminate this behavior.

Another point about our implementation is a bias towards paths on one side of the robot. Our robot has two stereo camera pairs, one pointing to the left, and one pointing to

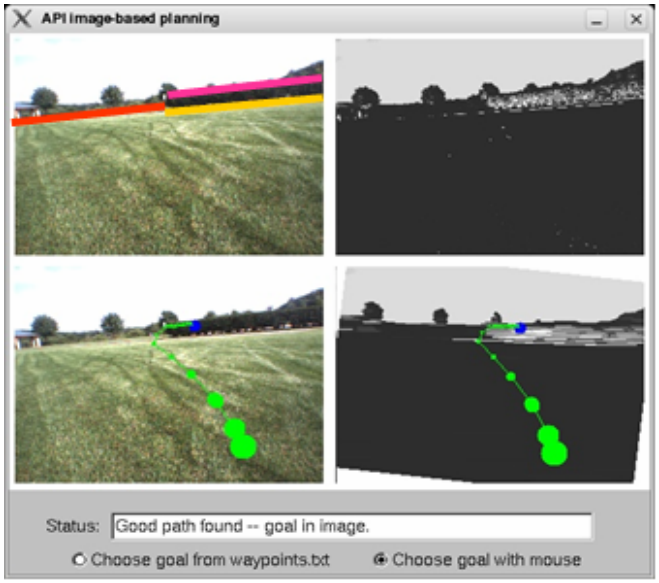

Fig. 10. Example of how surrounding terrain affects the ability of the image-based planner to plan around an obstacle.

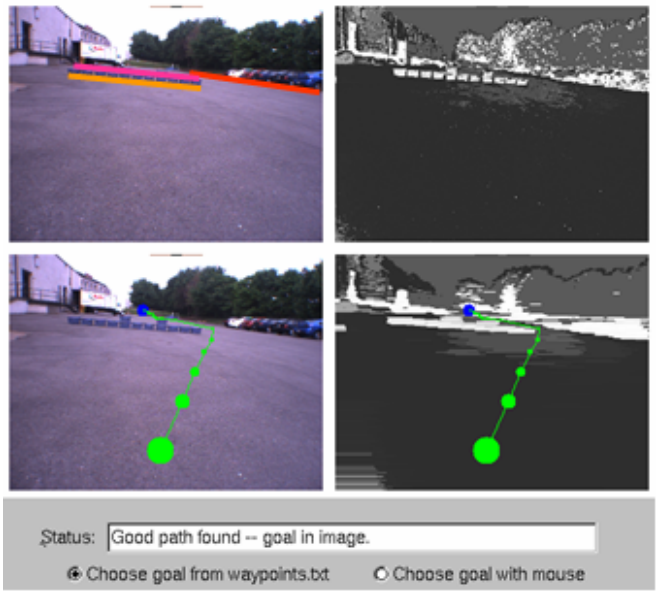

Fig. 11. Example of planning around a row of 0.4 m high plastic bins.

the right. We currently run image-based planning only on one camera. This may have the effect of biasing the robot to one side, but we did not have sufficient time to devise and test a method for resolving conflicts between image-based plans from cameras pointing in different directions. This problem could also be avoided by using an omnidirectional camera.

C. Effective planning range

One of the principal advantages of planning in image space is that it can enable a robot to react to obstacles far beyond traditional stereo range. The distance at which this planning is effective, however, depends on the size of the obstacle and on the nature of the surrounding terrain. For example, Figure 10 shows an example where the image-based planner was able plan around a 1.6 m tall hedge from over 20 meters away because the terrain to the left of the hedge, which includes some trees off in the distance, was classified as low-cost due to the color similarity with the grass in the foreground. Had this been classified as high cost, the image-based planner would have had trouble planning a path around the hedge. Figure 11 shows an example where a low-cost path almost all the way to the goal has been found.

So, effective range of our algorithm depends not just on obstacle size, but also on how much open space is visible around the obstacle.

V. Conclusions

We have presented a method for planning paths for a mobile robot in natural outdoor terrain directly in the image space of an on-board camera. This allows our system to react to obstacles at far greater ranges than the maximum range of our range sensors. We have described the details of our method, as well as how it is integrated into our system, working in conjunction with a Cartesian path planner. One attractive feature of our image-based planning algorithm is that its time and space complexity is dependent only on the size of the image which is constant for a given camera.

We have found image-based planning to provide its most useful guidance when avoiding distant obstacles in open terrain and when a clear path is visible in the image. It tends to be of limited use in cluttered environments, e.g., going through wooded areas without a trail. Our system has been independently tested through the DARPA LAGR and UPI programs, and has been shown to react to an obstacle at a range of 93 m, despite having reliable stereo data only to a range of 5 m.

Acknowledgments

This research has been supported through the DARPA LAGR program. The authors would like to thank Larry Jackel for discussions during the course of this work.

References

[1] L. D. Jackel, E. Krotkov, M. Perschbacher, J. Pippine, and C. Sullivan, "The DARPA LAGR program: Goals, challenges, methodology, and phase I results," *Journal of Field Robotics*, vol. 23, no. 11/12, pp. 945–973, 2006.

[2] H. Zhang and J. P. Ostrowski, "Visual motion planning for mobile robots," *IEEE Transactions on Robotics and Automation*, vol. 18, no. 2, pp. 199–208, April 2002.

[3] E. Rivlin, I. Shimshoni, and E. Smolyar, "Image-based robot navigation in unknown indoor environments," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2003, pp. 2736–2742.

[4] A. Howard, L. Matthies, A. Huertas, M. Bajracharya, and A. Rankin, "Detecting pedestrians with stereo vision: Safe operation of autonomous ground vehicles in dynamic environments," in *International Symposium on Robotics Research*, 2007, to appear.

[5] M. Happold, M. Ollis, and N. Johnson, "Enhancing supervised terrain classification with predictive unsupervised learning," in *Robotics: Science and Systems II*, 2006.

[6] M. Ollis, W. H. Huang, and M. Happold, "A bayesian approach to imitation learning for robot navigation," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2007, to appear.