Detecting Obstacles and Drop-offs using Stereo and Motion Cues for Safe Local Motion

Aniket Murarka
Department of Computer Science
The University of Texas at Austin
aniket@cs.utexas.edu

Mohan Sridharan
School of Computer Science
University of Birmingham, UK
mzs@cs.bham.ac.uk

Benjamin Kuipers
Department of Computer Science
The University of Texas at Austin
kuipers@cs.utexas.edu

***Abstract*— A mobile robot operating in an urban environment has to navigate around obstacles and hazards. Though a significant amount of work has been done on detecting obstacles, not much attention has been given to the detection of drop-offs, e.g., sidewalk curbs, downward stairs, and other hazards where an error could lead to disastrous consequences. In this paper, we propose algorithms for detecting both obstacles and drop-offs (also called negative obstacles) in an urban setting using stereo vision and motion cues. We propose a global color segmentation stereo method and compare its performance at detecting hazards against prior work using a local correlation stereo method. Furthermore, we introduce a novel drop-off detection scheme based on visual motion cues that adds to the performance of the stereo-vision methods. All algorithms are implemented and evaluated on data obtained by driving a mobile robot in urban environments.**

I. Introduction

Imagine a mobile robot driving across a university campus. In order to go from one building to another it would likely have to drive through corridors avoiding people and furniture (obstacles), take ramps or elevators instead of stairs, drive along sidewalks at a safe distance from the curb (a drop-off), and possibly cross streets. Thus, for a robot to move about autonomously in its environment it has to be able to detect and avoid obstacles and drop-offs (and other hazards), i.e., *safety* is essential. Table I categorizes some common hazards that a mobile robot can expect to find in an urban setting.

Hazards	**Examples**
Drop offs	Sidewalk curbs, downward stairs, steps
Obstacles: Static	Walls, furniture
Dynamic	People, doors
Invisible	Glass doors and glass walls
Overhangs	Table tops, railings, tree branches
Inclines	Wheelchair ramps, curb cuts
Rough surfaces	Gravel paths, grass beds
Narrow regions	Doorways, elevators

TABLE I: Common hazards in urban environments for a robot.

A lot of work in the mobile robotics literature has focused on detecting and avoiding obstacles [1], [2], with a significantly lesser amount of work done on detecting drop-offs and other hazards [3], [4], even though detecting such hazards is just as crucial. Laser range-finders have predominantly been the sensor of choice because they provide accurate range data with very little noise. Cameras are used less frequently, and as a secondary information source [2], because they are sensitive to environmental changes and the data they return is more difficult to interpret.

In this paper we focus on detecting drop-offs (also called negative obstacles in the literature [4]) and static obstacles in urban environments using vision-based algorithms. We introduce a color segmentation based global stereo algorithm and compare its performance at detecting hazards with a local correlation stereo algorithm. We also present a novel algorithm for detecting drop-offs using motion cues, and evaluate its performance in conjunction with the stereo methods. Overhanging objects are detected as well but (in this work) we do not distinguish them from static obstacles. Since an error can lead to disastrous consequences, it is essential for the robot to detect hazards with a high degree of accuracy and hence we experimentally measure the error rates of the algorithms.

We use cameras as the primary sensors instead of lasers, because cameras are significantly cheaper and smaller, and they provide much more information in a single frame of data. For the price of a single laser it is possible to mount several cameras on a robot providing a larger field-of-view that is important for safety. The smaller size of cameras also provides flexibility when mounting them on smaller mobile robots. Furthermore, the information from a camera can be used for purposes other than safety, e.g., object recognition.

The remainder of the paper is organized as follows. Section II provides an overview of the algorithms and our contributions while section III discusses related work. Section IV describes the stereo vision algorithms for obstacle and drop-off detection and section V introduces the motion cue based drop-off detection algorithm. Section VI describes the creation of the *local safety map*, from the output of stereo and motion based methods, to enable safe robot motion. The experimental setup and results are described in Section VII, followed by discussion and future work in Section VIII.

II. Overview of Methods and Contributions

We investigate the performance of three methods for drop-off and obstacle detection: two based on stereo vision, and one based on motion. The outputs of the methods are used to construct an annotated 2D grid map of the robot's *local* environment, called a *local safety map* [5]. Each cell in the safety map carries one of five labels indicating whether the corresponding region is an obstacle or overhang, a drop-off edge, a part of the ground plane, a region below the ground plane, or a region about which nothing is known. The safety map is local in nature and since it identifies safe regions in the robot's surroundings, it can be used by the robot for local path planning to avoid hazards [6]. We evaluate the methods by comparing local safety maps built by them against ground truth safety maps. The three methods are:

1. Correlation Stereo: A local stereo algorithm along with a noise removal process that we develop in earlier work [5] is used as the baseline. This algorithm is similar to other stereo-based obstacle detection algorithms.

2. Color Segmentation Stereo: Global stereo methods that perform a global optimization over the image, have been shown to give better depth estimates than local stereo methods [7], especially the set of methods that use color segmented images [8]. We draw on these approaches to develop our own global color segmentation stereo method for hazard detection - this results in maps with higher precision than the correlation stereo based maps.

3. Motion based Drop-off Detection: We also present a novel motion-based method that explicitly identifies drop-offs directly in front of the robot. Such drop-offs have an occluding edge and this method uses the relative motion (across several images) between this edge and other image features for detecting the drop-off.

The motion based method is designed to be used in conjunction with other obstacle detection methods, such as the stereo methods. It provides redundancy and robustness to the system in case the stereo method fails to detect a drop-off, and helps to locate the drop-off accurately.

In summary, to create a local safety map, we perform the following on each new frame of stereo images:

1. *Compute Depth Map:* The disparity (depth) map of the images is computed and converted from the camera's coordinate frame to the frame of the local safety map.
2. *Construct 3D Model by Removing Noise:* The transformed depth map is filtered to remove noise and obtain an updated 3D model of the robot's local environment.
3. *Compute Motion based Drop-off Edges:* Motion cues are used to find potential drop-off edges in one of the images, whose locations are mapped to the coordinate frame of the local safety map.
4. *Remove False Positive Drop-off Edges:* The motion based drop-offs are accumulated in a 2D grid to reduce the number of false positives.
5. *Construct Local Safety Map:* The stereo based 3D model stereo is projected on a 2D grid and merged with the 2D motion drop-off grid to get the current local safety map.

In this work, we assume that: (i) We have an independent process that is able to localize the robot in its *local* surroundings. We use a laser based 3-DOF local SLAM algorithm [9] but it can be replaced by a camera based SLAM [10] or structure from motion algorithm [11]. (ii) The robot travels only on level (horizontal) surfaces, though the constraint can be relaxed by using a 6-DOF visual localization module. (iii) The robot's environment can be modeled using planes. Since we focus on urban environments this holds to a good degree (see Sec. IV-B). (iv) We have good lighting - this is required for the vision methods to operate properly.

III. Related Work

Here we discuss some representative prior work on hazard detection using vision and other sensors. Laser rangefinders are used extensively for detecting obstacles and other hazards. In their DARPA Grand Challenge vehicle, Thrun et al. [2] use multiple lasers mounted on top of the car to construct a local 3D point cloud of the regions in front of the car. Heckman et al. [4] find drop-offs using 3D laser data. They ray trace to find occlusions in the 3D laser grid and then determine the cause of the occlusions. Wellington, et al. Wellington et al. [12] use lasers and cameras to find the true ground height and hence traversability of vegetation-covered regions. All these methods, however, require the use of expensive laser sensors.

Stereo vision based methods are quite commonly used for obstacle detection and navigation. Gutmann et al. [6] propose a stereo vision-based navigation system for humanoid robots, using a 2.5D grid to represent the world, each grid cell having a height and a label (floor/obstacle). To get accurate floor heights, the raw range data is segmented into planes. Singh et al. [13] fit planes to stereo data to construct 2D local grid maps, and to estimate the traversability of cells. These methods do not use color cues for segmentation or perform a global optimization which can lead to improved performance.

Two vision-based methods for the detection of drop-offs are evaluated in [14]. One method looks for gaps in stereo range data, while another looks at height differences and gaps in local terrain/height maps of the environment. The results from both methods are combined to identify drop-offs. The stereo method is local and does not perform a global optimization to compute depth. Rankin et al. [3] merge stereo vision and thermal signatures to detect drop-offs at night, requiring the use of special infrared cameras.

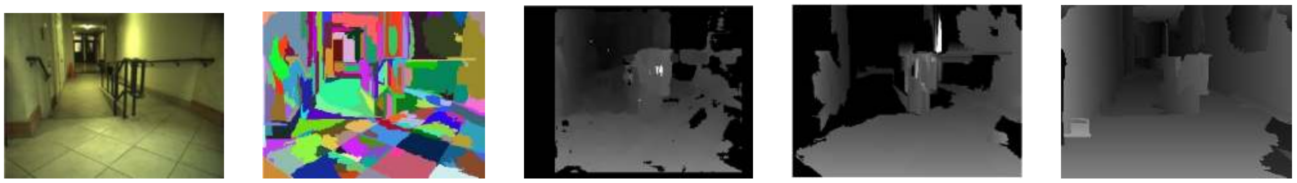
Related to our motion-based drop-off detection method, Stein and Hebert [15] propose an optical flow-based occlusion detection method to detect occlusions in videos with moving objects. Instead of optical flow, we measure the *relative* motion of features with respect to a fixed edge (this works even when objects do not move). Relative motion is observed to be a more sensitive measure, and in combination with stereo vision-based methods it results in a robust drop-off and obstacle detection method.

IV. 3D Reconstruction using Global Stereo

Here we describe the process of 3D scene reconstruction using our proposed color segmentation based global stereo algorithm. We compare the results of this method against an earlier local stereo based reconstruction algorithm (see [5] for details). The output from both methods is used to construct local 2D safety maps (section VI) which are then evaluated against ground truth safety maps to determine the performance of both methods in section VII. We begin with the coordinate transformations used in the reconstruction, which provide the location of points in the coordinate frame of the local safety map.

A. Coordinate Transformations

We assume that we have a calibrated stereo camera with known parameters. We can express a point's 3D coordinates in the camera's coordinate frame $\mathbf{x}^c = (x^c, y^c, z^c)^T$ as a known function f of the point's image coordinates (column, row, and



(a) Left image of stereo pair. (b) Color segmented left image. (c) Correlation stereo disparity map of left image. (d) Disparity map after fitting world planes. (e) Disparity map after energy optimization.

Fig. 1: Images from various stages of the global color segmentation stereo algorithm.

disparity) $\mathbf{x}^i = (c, r, d)^T$: $\mathbf{x}^c = f(\mathbf{x}^i)$. The camera coordinates can be transformed to yield the point's location in the robot's coordinate frame: $\mathbf{x}^r = R_c\mathbf{x}^c + T_c$, where R_c and T_c are known rotations and translations relating the camera frame to the robot frame. Since localization gives the robot's pose in the coordinate frame of the local safety map (the global frame), the point's 3D position in the global frame can be computed: $\mathbf{x}^g = R_r\mathbf{x}^r + T_r$.

B. 3D Reconstruction

Global stereo methods claim to give significantly better depth estimates than local stereo methods [7]. We therefore build on recent work on color segmentation stereo [8] to develop a stereo algorithm for detecting hazards.

1) Color Segmentation based Stereo: Color segmentation algorithms work by first segmenting an image into segments of homogeneous color and then computing the disparity of each segment. Based on the premise that significant disparity discontinuities do not occur inside a region of homogeneous color, most methods fit a disparity plane to each color segment in one stereo image. Since the assumption of planar color segments is not always true, most approaches tend to over-segment the image to better approximate the true disparity [8]. Over-segmentation allows non-planar surfaces to be approximated using several small planes. We believe that for urban environments, which are mostly composed of planar or smooth surfaces, this approximation should work fairly well. The final step of global segmentation methods then involves minimizing a global cost function that measures the quality of fit over the entire image and adherence to different constraints (e.g. smoothness). The main steps in our algorithm are as follows (shown in Fig.1).

(i) Color Segmentation. The left stereo image (Fig. 1(a)) is color-segmented (Fig. 1(b)) using the algorithm in [16].

(ii) Initial Disparity Computation. The left image disparity map (Fig. 1(c)) for the stereo images is computed using a stereo algorithm available from [17].

(iii) Fitting World Planes to Segments. Instead of fitting planes in disparity space to each segment as in other color segmentation algorithms [8], we fit planes in Euclidean space to handle errors better. We compute the 3D coordinates, in the robot's reference frame, of all pixels in a segment with a valid disparity using the equations in section IV-A. Then we fit either a vertical plane ($q_1x + q_2y + q_3 = 0$) or a horizontal plane ($z + q_3 = 0$) to the 3D points (Fig. 1(d)), using a weighted least squares method [8]. The goal of this step is to get a set of planes representative of the major planes in the robot's environment. The following steps describe how the "correct" plane is chosen for each segment.

(iv) Adding Additional Planes. In order ensure that the candidate plane set contains all major horizontal planes in the robot's environment, several pre-calculated evenly distributed horizontal planes are added, e.g., all horizontal planes between -2 and 2 meters that are 0.1 meters apart. This helps find correct ground and below ground planes.

(v) Refining the Set of Planes. The plane set, which now contains several similar planes, is refined by forming clusters of planes based on height, orientation, and distance from origin, and then picking the best plane from each cluster. This refinement significantly reduces the number of planes and provides smoother depth maps. In order to evaluate the quality of planes in a given cluster we first convert the world planes to disparity planes of the form:

$$d = p_1c + p_2r + p_3 \tag{1}$$

For each such disparity plane P we compute the difference in intensities, for the disparity hypothesis given by the plane, at every pixel (r, c) in the image, i.e., $\Delta I_P(r, c) = |I_L(r, c) - I_R(r, c - d)|$. $I_L(r, c)$ and $I_R(r, c)$ are the left and right image intensities at (r, c) (for color images the RGB intensities at each pixel are added to compute $\Delta I_P(r, c)$). The total number of pixels for which $\Delta I_P(r, c)$ is below a threshold, then gives the *support* of the plane in the image. The best plane in each cluster is the plane with maximum support.

(vi) Energy Minimization for Selecting Correct Planes. In the final step, the set of disparity planes that minimize a global energy function over the two images are chosen as the "correct" planes. To define the energy function we first define a matching cost for a given disparity plane P and segment S: $C(S, P) = \sum_{(r,c)\in S} |I_L(r, c) - I_R(r, c - d)|$, where d is computed using Eqn. 1.

We also define a labeling f that assigns a plane to every segment S, i.e., $f(S)$ is the plane corresponding to S for a given labeling f. The energy function corresponding to a labeling f of an image is:

$$E(f) = \sum_S C(S, f(S)) + \sum_{S,S'} L_{S,S'}\delta(f(S) \neq f(S')) \tag{2}$$

where the first sum is over all segments and the second sum is over all pairs of neighboring segments. $L_{S,S'}$ is proportional to the boundary length between segments S and S' and $\delta(f(S) \neq f(S'))$ is 1 when its argument is true otherwise 0. Minimizing this energy function gives us the optimal labeling f for the image which in turns gives the "correct" disparity plane for each segment S. The initial labeling for starting the optimization is chosen by minimizing only $C(S, P)$ for each segment independently. We use code available online and methods described in [18], [19], [20], [21] to find a strong local minima of the energy function. For more details the reader is also referred to [8].

Fig. 1(e) shows the final disparity map obtained after finding the "correct" labeling. Segmentation stereo hence returns a set of segments, each associated with a vertical or horizontal plane that gives the depth of all pixels in the segment.

2) Noise Removal and 3D Reconstruction: Unfortunately, the depth estimates obtained from the segmentation stereo algorithm are noisy and contain many false positives, e.g., when the wrong plane is found for a segment. Noise reduction is achieved by accumulating depth data returned by segmentation stereo over several frames in two 3D grids, one each for horizontal and vertical planes. The grid origins are defined with respect to the global coordinate frame and they are on the order of $10m \times 10m \times 4m$ in size (in x, y, and z respectively) representing a small local space. We use a cell size of $0.05m^3$, with all cells initialized to zero.

The value of each cell through which a plane passes is incremented by a fixed amount (each plane contributes once to a cell). Only cells with values above a certain threshold are considered occupied., thereby reducing noise and removing false positives. An obvious drawback of this method is that over a period of time each cell in the grid will eventually become occupied. This can be handled by having cell values "decay" over time, or by incorporating negative evidence which is part of future work.

Information passed on to the Safety Mapper: The safety mapper, described in section VI, uses both horizontal and vertical grids at each time-step for computing the safety map.

V. Drop-off Detection with Motion Cues

Unlike the stereo-based methods which compute a disparity map to detect hazards, drop-offs in front of the robot can be explicitly detected using motion cues. Drop-offs on the ground surface will have an occluding edge. As the robot moves towards the edge, regions occluded by the edge will come in view, and when the robot moves away from the edge, regions that are initially visible will slowly disappear. Algorithm 1 lists the operations performed on the video stream to detect drop-offs. Figure 2 shows image results at various stages of Algorithm 1, when applied on a pair of images separated by N (=5) frames, in an outdoors environment. For ease of explanation we assume the robot is moving towards a drop-off.

The underlying principle is that, for an occluding edge, new regions come into view as the robot moves towards the edge. For an occluding edge, features above the edge will hence appear to move "faster" relative to the edge than features below the edge, as compared to a non-occluding edge. A threshold on δ_{ab} can be used to distinguish between occluding and non-occluding edges (Fig. 2). To make the method relatively insensitive to the threshold, we set the threshold such that the motion algorithm detects potential drop-offs aggressively and has a high false positive rate.

Algorithm 1 Motion-based drop-off detection.

```
Require: A pair of images, $I_1$, $I_2$ separated by $N = 5$ frames.
1: Find all edges in images $I_1$, $I_2$ – we use Peter Kovesi's MATLAB functions [22].
2: Only consider edges that are (i) nearly horizontal in the image (slope $\leq 30^o$) and (ii) not very far off (distance $\leq 20m$).
3: Match edges in $I_1$ with edges in a previous image, $I_2$. The separation of five frames ensures appreciable motion thereby increasing the signal-to-noise ratio. Let $M$ = number of matched edges.
4: for i = 1 to $M$ do
5:     Find invariant features [23] above and below edge $E_i$ in $I_1$ and $I_2$, within a certain region around the edge.
6:     Match the features across $I_1$ and $I_2$ – features above/below the edges are matched separately.
7:     Compute the distances in pixels to $E_i$ ($d_{a_1}$, $d_{b_1}$, $d_{a_2}$, $d_{b_2}$), of matching features above and below $E_i$ in the corresponding images. Histogram the vectors and pick the bin with maximum count as the corresponding mean distance: $d_{a_{1m}}$, $d_{b_{1m}}$, $d_{a_{2m}}$, $d_{b_{2m}}$.
8:     Measure the movement of features above and below edge $E_i$ between $I_1$, $I_2$, i.e. $\delta_a = |d_{a_{1m}} - d_{a_{2m}}|$, $\delta_b = |d_{b_{1m}} - d_{b_{2m}}|$.
9:     if $\delta_{ab} = (\delta_a - \delta_b) >$ Threshold then
10:        $E_i$ is an occluding edge.
11:    else
12:        $E_i$ is a non-occluding edge.
13:    end if
14: end for
```

Algorithm 1 can only detect frontal drop-offs and not lateral drop-offs. Since the method does not know which occluding edges are actually on the ground plane, it treats all occluding edges as potential drop-offs, e.g., in Fig. 2 last row, the occluding edge of a pipe in the distance is also treated as a potential drop-off. This method is hence combined with stereo methods capable of finding the ground plane.

Noise removal and information passed on to the Safety Mapper: For a given image, algorithm 1 finds a set of potential drop-off edges. Since drop-offs are assumed to be on the ground plane ($z = 0$), the 3D location of edge pixels can be computed using equations in section IV-A, setting z^r to zero, and solving for disparity d. The ground plane drop-off locations are then accumulated on a 2D grid – each cell through which a drop-off edge passes is incremented by a fixed amount. Only cells with values above a threshold are marked as being potential drop-offs. The process thus handles the high number of false positives. This 2D motion drop-off grid is passed onto the safety mapper at every time-step for computing the safety map.

VI. Creating the Local Safety Map

The safety map is a 2D grid of fixed size with each cell annotated with one of five labels: *ground, below ground, above ground, unknown, or drop-off edge*, indicating the robot's

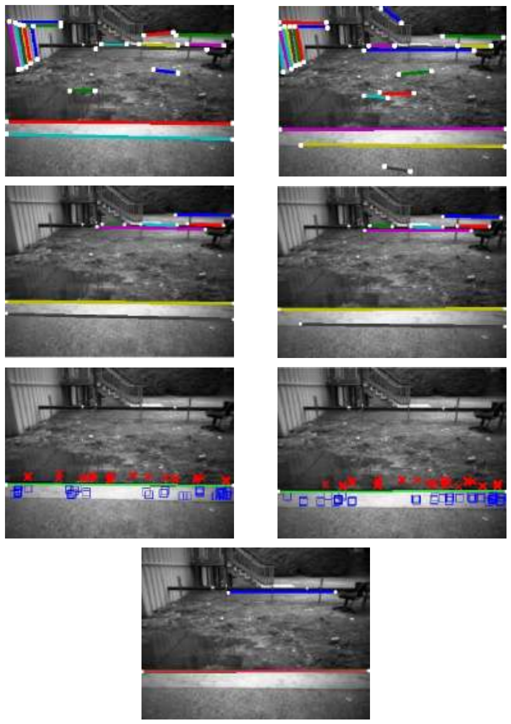


Fig. 2: Algorithm 1 applied to an outdoor environment (better viewed in color). 1st Row (Step 1 of Algorithm 1) : Edge detection on the image pair. 2nd Row (Steps 2 to 3): Edges are filtered and matched across the image pair. 3rd Row (Steps 4 to 8): For a pair of matched edges, features are found and matched across the image pair, above and below the edge separately. The distance moved by the matched features, above and below the edge, is computed separately. 4th Row (Steps 9-13): Drop-offs found.

current state of knowledge about the region corresponding to the cell. Regions labeled ground are at ground level and considered safe; above ground regions correspond to obstacles or overhangs and are unsafe; regions below ground indicate that a drop-off may be present at the boundary between them and ground regions; drop-off edges are unsafe; unknown regions are also to be avoided. The safety mapper merges information obtained from the two stereo methods and the motion based drop-off detector, to construct a safety map as follows:

1. Information from Stereo: The segmentation stereo method returns horizontal and vertical 3D grids with occupied cells marked. The 3D grids are projected onto a 2D grid to get the safety map. A 2D cell in the safety map is marked as: (i) *above ground*, if all corresponding occupied 3D cells are above a height of 0.1 meters; (ii) *on ground*, if most of the corresponding occupied 3D cells are between -0.1 m and 0.1 m; (iii) *below ground* if most corresponding occupied 3D cells are below -0.1 m; (iv) *unknown* if no corresponding 3D cells are occupied. A similar process is followed for the 3D point cloud returned by correlation stereo.

2. Information from Motion based Drop-offs: The motion drop-off method returns a 2D grid with potential drop-off cells marked. This grid is combined with a safety map, created using one of the stereo methods, to obtain a combined stereo and motion safety map. In the combined safety map, cells in the stereo safety map marked *on ground* are re-annotated as *drop-off edges* if the corresponding cells in the motion grid are marked as drop-off edges.

VII. Experimental Evaluation and Results

Fig. 3 shows our research platform, a robot wheelchair with two laser range-finders, one mounted horizontally and the other vertically, and a stereo camera mounted on a pan-tilt unit. All sensors have been calibrated against each other.

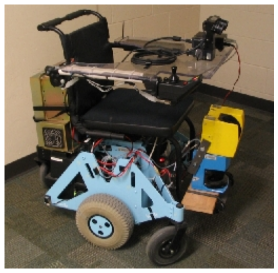


Fig. 3: Wheelchair Robot

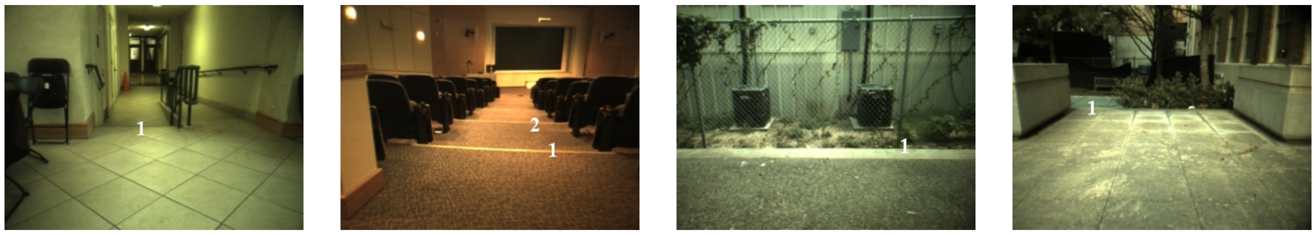
The algorithms are evaluated on four stereo video data sets (on the order of 500 stereo image pairs each) collected by driving the robot through two indoor and two outdoor environments shown in Fig. 4. The following algorithms are evaluated: (A) MD: motion based drop-off detection. (B) CS: correlation stereo. (C) SS: segmentation stereo. (D) CS+MD: correlation stereo combined with the motion based drop-off detection. (E) SS+MD: segmentation stereo combined with the motion based drop-off detection. We compare the hazard detection accuracy, detection distance and latency, and running times of the algorithms.

1) **Hazard Detection Accuracy**: Algorithms (B) through (E) were evaluated by comparing the safety maps built by them against ground-truth safety maps that were obtained by manually annotating and "cleaning" the safety maps created using the robot's horizontal and vertical laser range-finders. We compute the true positive (TP), true negative (TN), false positive (FP), false negative (FN) rates, and precision (PR) and recall (RC) of each algorithm. The error rates are computed on a per cell basis for the safety maps: (i) TP: % of unsafe cells marked unsafe, (ii) TN: % of safe cells marked safe (iii) FP: % of safe cells marked unsafe, (iv) FN: % of unsafe cells marked safe, (iv) PR: % of all map cells marked unsafe that are actually unsafe, and (v) RC: % of all actually unsafe map cells that are marked unsafe. Table II presents the results of algorithms (B) to (E) averaged over all four environments.

	CS	SS	CS+MD	SS+MD
TP	84	71	84	71
TN	73	82	72	82
FP	18	12	20	13
FN	7	5	7	6
PR	81	92	81	91
RC	93	93	93	93

TABLE II: Error rates of algos. averaged across four environments.

Algorithm (A) was evaluated based on the 2D motion grids it created for all environments. Instead of on a per cell basis the above metrics were computed on a per drop-off basis. The results for algorithm (A) are as follows: In all environments, MD detects all 5 frontal drop-offs present, corresponding to TP=100. Correspondingly, since it never fails to detect a frontal drop-off, so FN=0. However, MD also returns 7 false



(a) "Ramp" environment. (b) "Auditorium" environment. (c) "Fence" environment. (d) "Plants" environment.

Fig. 4: Sample images from the four video data sets collected on our University campus for which safety maps are constructed. The numbers show the location of drop-off edges in each image and are used to cross-reference with Fig. 5.

positives. Unfortunately, it is very difficult to accurately find the total number of edges that the MD algorithm examined - we estimate there were between 10-30 such edges in each environment. Hence we cannot calculate the FP (and TN) error *rate(s)* and have presented the FP error itself. Figure 5 shows the safety maps and motion grid maps created using the methods for different environments. We observe that:

- CS has a higher TP rate than SS, i.e., CS is able to *cover* the environment better. However, SS does better in *all other* metrics - it has lower FP and FN rates.
- Both stereo based methods have high recall. However, SS has higher precision and this is seen in the "cleaner and crisper" maps that SS produces in Fig. 5(b) as compared to CS in Fig. 5(c).
- The MD algorithm detects all drop-offs even at the cost of finding some false positives. As seen in Fig. 5(h), MD detects very small drop-offs ($\approx 10-15$cm high in Fig. 4(b)), with one of them at a large distance from the camera. It also provides the exact locations of drop-offs, unlike CS/SS, thereby increasing the robustness of the stereo methods.
- The numbers for CS+MD and SS+MD are very similar to those for CS and SS respectively, which is to be expected as the drop-offs occupy a small portion of the overall environment (see Fig. 5).

Based on the observations we cannot conclusively say that the segmentation stereo based method does better than the correlation stereo based method, however, it is clear that the segmentation stereo method produces much crisper maps. In effect future work will focus on improving the true positive (TP) rate of segmentation stereo so as to get the best of both methods. The motion method does very well at detecting frontal drop-offs and combined with either of the stereo methods is very useful for accurately locating drop-offs.

2) **Hazard Detection Latency and Distance:** All three methods (CS, SS, MD) produce noisy estimates, and in the absence of some noise filtering, the robot may very well be "paralyzed with fear". But, filtering involves accumulating evidence over several frames, leading to latencies in hazard detection. We present representative results of measuring the latencies and the distances at which hazards are detected, from which time-to-collision and velocity constraints on the robot can be computed.

Specifically: (i) *Latency* is the number of camera frames between the appearance of a hazard in a video sequence, and its detection by the robot. (ii) *Detection distance* is the distance to a hazard when it is first detected. CS methods have a minimum algorithmic latency of 7 frames. In the "Ramp" environment (Fig. 4(a)), CS detects the left wall and railing after 7 frames at distances of ≈ 2.8m and 3m respectively. SS methods have a minimum algorithmic latency of 16 frames. In the "Ramp" environment, the left wall and the region beyond the drop-off edge were detected in ≤ 20 frames at a distance of 2.5m and 4m respectively. MD also has an algorithmic latency of 16 frames, and in the "Ramp" environment the algorithm detected the exact drop-off location after 25 frames. Improving the filtering schemes can reduce the latency.

3) **Time Analysis:** On a machine with a 2.13 GHz dual core processor, the segmentation stereo based method (including noise removal and safety map creation) takes on average 4.5 sec per stereo frame. Most of the time is spent computing the disparity maps with other times being negligible. The code is fairly optimized and written in C++ and Matlab. The correlation stereo based method takes 2 sec on average. Noise removal takes up most of the time with disparity map computation taking negligible time. The code is fairly well optimized. The motion based method takes 6.7 sec on average (un-optimized Matlab code).

VIII. Discussion and Future Work

Mobile robots navigating autonomously in an urban environment need a mechanism to detect obstacles and other hazards. Unlike previous work that has predominantly used non-visual sensors and focussed on obstacles, we present vision-based algorithms for detecting drop-offs and static obstacles.

Two methods for hazard detection are proposed: a new global color segmentation stereo algorithm (SS), and a novel motion-based drop-off detector (MD). The segmentation stereo method is compared to a correlation stereo based algorithm (CS) based on our prior work [5]. The safety maps generated by each of the three methods are compared against ground-truth safety maps obtained by manually annotating laser-based range maps. The stereo methods perform comparably, with the correlation stereo method having a higher true positive rate but the segmentation based method doing better on all other metrics, in particular precision, resulting in "crisper" local safety maps. These results suggest that color segmentation based methods have the potential to perform very well. Though the addition of MD does not provide significant quantitative difference (drop-offs are a

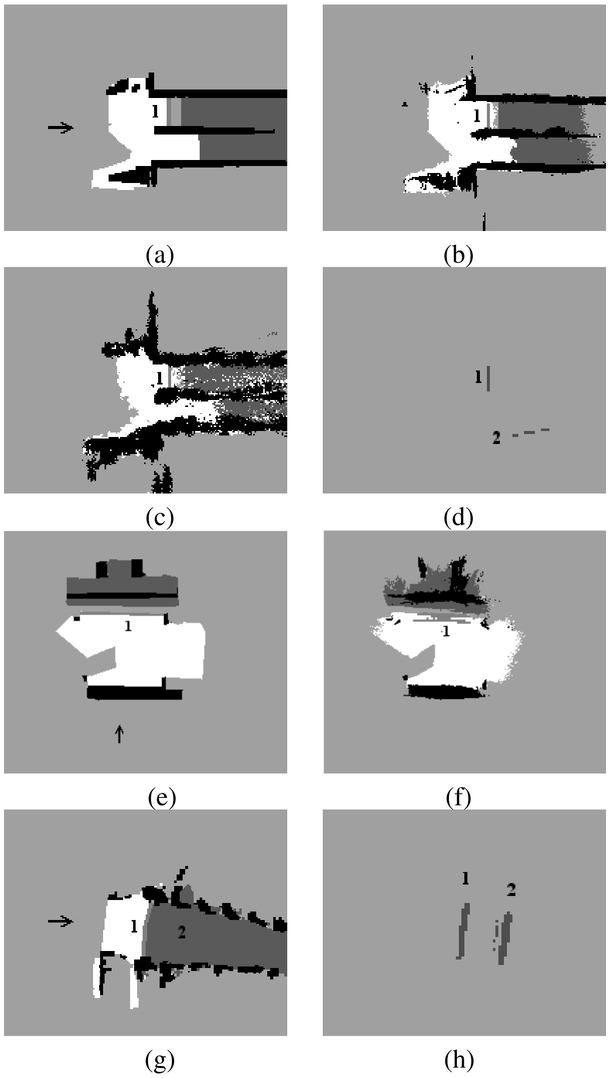


Fig. 5: The figures show safety maps and motion based drop-off grids created for various environments. Figures (a,b,c,d) are for the "Ramp" environment. Fig. (a) shows the environment's ground truth safety map. The arrow shows the direction from which the image in Fig. 4(a) was taken (similarly for figures (e) & (g)). This safety map and others are annotated as follows: White: on ground (safe); Black: above ground (unsafe); Dark gray: below ground (unsafe); Light gray: unexplored spaces; Intermediate gray: drop-off edges (with a number next to it). Fig. (b) shows the safety map (SS+MD) created by combining the segmentation stereo safety map and and motion based drop-off grid shown in (d). Fig. (c) shows the combined safety map (CS+MD) created using correlation stereo and motion based methods. Fig. (d) shows that the motion based method accurately finds drop-off location #1 and also returns one false positive drop-off #2. Fig. (e) & (f) show the ground truth and the SS+MD safety maps for the "Fence" environment respectively. The fence causes the segmentation stereo method some trouble but is nevertheless detected. Fig. (g,h) are for the "Auditorium" environment. Fig. (g) is the ground truth safety map showing two drop-offs. Both drop-offs are detected by the motion based method in (h) even though they are only 10-15cm high.

small portion of the environments), it results in accurate localization of the drop-offs even for small drop-offs. We thus show that it is possible to reliably detect hazards using the rich information encoded in visual data.

In the future, we plan to extend this work to non-level environments by using a 6DOF localization module. Latencies can be reduced by incorporating a more sophisticated noise filtering process. We also aim to generalize segmentation stereo by considering planes at all orientations. Eventually, we aim to enable a mobile robot to navigate autonomously (and safely) in urban environments.

IX. Acknowledgments

This work has taken place in the Intelligent Robotics Lab at the Artificial Intelligence Lab, The University of Texas at Austin. Research of the Intelligent Robotics lab is supported in part by grants from the Texas Advanced Research Program (3658-0170-2007), from the National Science Foundation (IIS-0413257, IIS-0713150, and IIS-0750011), and from the National Institutes of Health (EY016089).

References

[1] I. Ulrich and I. Nourbakhsh, "Appearance-based obstacle detection with monocular color vison," in *AAAI*, 2000.

[2] S. Thrun, et al, "Stanley: The Robot that Won the DARPA Grand Challenge," *Journal of Field Robotics*, 2006.

[3] A. Rankin, A. Huertas, and L. Matthies, "Nighttime negative obstacle detection for off-road autonomous navigation," in *SPIE*, 2007.

[4] N. Heckman, J. Lalonde, N. Vandapel, and M. Hebert, "Potential negative obstacle detection by occlusion labeling," in *IROS*, 2007.

[5] A. Murarka, J. Modayil, and B. Kuipers, "Building Local Safety Maps for a Wheelchair Robot using Vision and Lasers," in *The Third Canadian Conference on Computer and Robot Vision*, 2006.

[6] J.-S. Gutmann, M. Fukuchi, and M. Fujita, "Real-time path planning for humanoid robot navigation," in *IJCAI*, 2005.

[7] D. Scharstein and R. Szeliski, "A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms," *IJCV*, 2002.

[8] L. Hong and G. Chen, "Segment-based Stereo Matching using Graph Cuts," in *CVPR*, 2004.

[9] S. Thrun, W. Burgard, and D. Fox, "A real-time algorithm for mobile robot mapping with applications to multi-robot and 3D mapping," in *IEEE Intl. Conf. on Robotics and Automation*, 2000.

[10] A. Comport, E. Malis, and P. Rives, "Accurate quadrifocal tracking for robust 3d visual odometry," in *ICRA*, 2007.

[11] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2000.

[12] C. Wellington, A. Courville, and A. T. Stentz, "Interacting Markov Random Fields for Simultaneous Terrain Modeling and Obstacle Detection," in *Robotics: Science and Systems*, June 2005.

[13] S. Singh, R. Simmons, T. Smith, A. Stentz, V. Verma, A. Yahja, and K. Schwehr, "Recent Progress in Local and Global Traversability for Planetary Rovers," in *ICRA*, 2000.

[14] A. Rankin, A. Huertas, and L. Matthies, "Evaluation of Stereo Vision Obstacle Detection Algorithms for Off-Road Autonomous Navigation," in *32nd AUVSI Symposium on Unmanned Systems*, 2005.

[15] A. Stein and M. Hebert, "Local detection of occlusion boundaries in video," in *BMVC*, 2006.

[16] D. Huttenlocher and P. Felzenszwalb, "Efficient graph based image segmentation," *IJCV*, vol. 59, no. 2, pp. 167–181, 2004.

[17] "Videre Design," http://www.videredesign.com/.

[18] Y. Boykov, O. Veksler, and R. Zabih, "Efficient Approximate Energy Minimization via Graph Cuts," *PAMI*, 2001.

[19] V. Kolmogorov and R. Zabih, "What energy functions can be minimized via graph cuts?" *PAMI*, 2004.

[20] Y. Boykov and V. Kolmogorov, "An Exp. Comp. of Min-Cut/Max-Flow Algorithms for Energy Minimization in Vision," *PAMI*, 2004.

[21] S. Bagon, "Matlab Wrapper for Graph Cut," 2006, http://www.wisdom.weizmann.ac.il/ bagon.

[22] "Peter Kovesi's Image Processing Matlab Functions," http://www.csse.uwa.edu.au/ pk/research/matlabfns/.

[23] K. Mikolajczyk and C. Schmid, "Scale and affine invariant interest point detectors," *IJCV*, vol. 60, no. 1, pp. 63–86, 2004.