Descending-stair Detection, Approach, and Traversal with an Autonomous Tracked Vehicle

Joel A. Hesch, Gian Luca Mariottini, and Stergios I. Roumeliotis

***Abstract*— This paper presents a strategy for descending-stair detection, approach, and traversal using inertial sensing and a monocular camera mounted on an autonomous tracked vehicle. At the core of our algorithm are vision modules that exploit texture energy, optical flow, and scene geometry (lines) in order to robustly detect descending stairwells during both far- and near-approaches. As the robot navigates down the stairs, it estimates its three-degrees-of-freedom (d.o.f.) attitude by fusing rotational velocity measurements from an on-board tri-axial gyroscope with line observations of the stair edges detected by its camera. We employ a centering controller, derived based on a linearized dynamical model of our system, in order to steer the robot along safe trajectories. A real-time implementation of the described algorithm was developed for an iRobot Packbot, and results from real-world experiments are presented.**

I. Introduction

Enabling robots to transition from the structured environments of laboratories and factory floors, to *semi-structured* urban and domestic environments, which contain steps and stairs is still an open problem. Existing approaches for autonomous robotic *stair navigation* provide only partial solutions. For instance, some only address the aspect of stair detection [2], while others only address control [3]. The vast majority of the available methods are limited to ascending stairs in a carefully controlled environment, e.g., with constant lighting, color-coded stairs [4], or known stair dimensions [5]. The problem of vision-based *stair descending* is particularly difficult, due to the challenge of identifying and localizing a descending staircase in an unknown environment using only visual cues. In addition, descending-stair traversal for autonomous tracked vehicles is challenging since track slippage can lead to the robot toppling off the stairs.

In this paper, we present a strategy for descending-stair detection, approach, and traversal for an autonomous tracked vehicle. To the best of our knowledge, this is the first work to explicitly examine the more difficult case of detecting and navigating descending staircases. Specifically, we focus on the minimum-sensing scenario in which only a monocular camera and a tri-axial gyroscope are available on the robot. Our algorithm is divided in four phases (see Fig. 1): (P1) *Far-approach:* the robot exploits texture-based energy measures to determine possible descending-stair locations and starts to navigate towards one of them. (P2) *Near-approach:* the robot verifies the hypothesis using optical-flow analysis of the leading stair edge. (P3) *Stair alignment:* the robot aligns to the descending stairs. (P4) *Stair traversal:* the robot steers along a safe trajectory down the descending staircase.

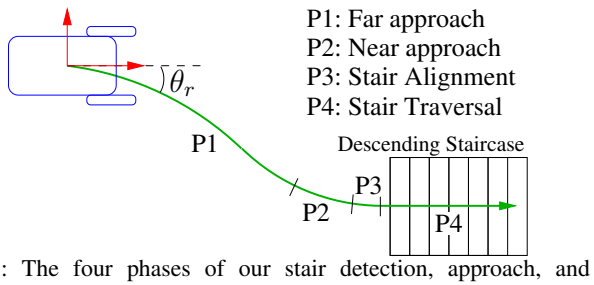

Fig. 1: The four phases of our stair detection, approach, and traversal strategy: P1: descending-stair candidate locations are determined, and the robot follows reference heading θ_r, towards one hypothesis. P2: optical flow is exploited to determine the precise stair location. P3: the robot aligns to the stairs. P4: the robot descends the stairs and then leaves the staircase.

In order to estimate the three-degrees-of-freedom (d.o.f.) attitude of the robot during P4, we employ an Extended Kalman Filter (EKF) that fuses rotational velocity measurements from a tri-axial gyroscope with image observations of stair edges from the monocular camera. Additionally, we use a PID controller to steer the robot along desired trajectories, both while the robot is on the stairs, and when the robot is approaching the stairs on flat ground [6]. Together, our perception, estimation, and control strategies provide a robust solution to the challenging problem of descending-stair detection, approach, and traversal.

The rest of the paper is organized as follows: In Sect. II, we discuss related work on stair detection and traversal. We present an overview of our strategy in Sect. III, and in Sect. IV we describe each algorithmic component in detail. Lastly, we present our experimental validation using an iRobot Packbot (Sect. V), as well as our concluding remarks and future work (Sect. VI).

II. Related Work

A. Stair Perception

Existing methods for stair detection have focused primarily on the task of identifying *ascending* staircases from laser or camera data. Stair detection has been achieved by

This work was supported by the University of Minnesota (DTC), and the National Science Foundation (IIS-0643680, IIS-0811946, IIS-0835637). The authors would like to thank Dr. Thomas Brox for providing the binary implementation of the variational optical flow method presented in [1], and Dr. Nikolas Trawny for his invaluable support during the document preparation process.

J. A. Hesch and S. I. Roumeliotis are with the Dept. of Computer Science and Engineering, University of Minnesota, Minneapolis, MN 55455, USA {joel|stergios}@cs.umn.edu
G. L. Mariottini is with the Dept. of Computer Science and Engineering, University of Texas, Arlington, TX 76019, USA
gianluca@uta.edu

appropriately engineering the environment (e.g., detecting color-coded steps with a stereo camera [4]), or by limiting the detection process to stairs of approximately known dimensions (e.g., known height [5]). Stair-detection based on Gabor filtering of monocular camera images has also been proposed [2]. These approaches benefit from the prominent appearance of the ascending staircase in the sensor data. However, to the best of our knowledge, there exists no work that explicitly addresses the far more difficult problem of detecting *descending* staircases from monocular images.

B. Stair Traversal

The task of stair ascending using tracked robots has been investigated by several researchers. Vu et al. [7] designed a four-tracked robot, which climbs stairs by executing a sequence of alternate rear and front track rotations. Although the authors model the tread depth and riser height, the robot's attitude is not estimated, therefore no heading corrections can be computed during the ascent. In [8], a tracked robot is equipped with a suite of sensors (i.e., sonar, monocular camera, and two-axis accelerometer) to estimate its orientation while on the stairs. However, their approach does not fuse all available sensor measurements, but instead uses heuristics to select the most accurate sensor. Still other approaches exist which utilize only monocular vision [9], or a combination of vision and gyroscopes [10] to estimate the orientation of the robot on the stairs. However, both [9] and [10] relied on the limiting assumption that the robot has zero-roll angle and constant pitch while on the stairs.

The predecessor to our current work [6], employed a tightly-coupled vision-aided inertial navigation system (V-INS) to estimate the three-d.o.f. attitude of the vehicle as it climbed the stairs, based on the gyroscope measurements and monocular observations of the stair edges. For controlling its motion, the robot switched between two heading strategies while on the stairs: (i) when near the unsafe stair-edge zone, it steered towards the middle of the stair, and (ii) when it was in the middle of the stair, it steered strait up the stairs.

In the current work, we build upon the estimation and control framework presented in [6], to enable new capabilities for autonomous descending-stair detection, approach, and traversal. Our problem is significantly more challenging, since the descending staircase is not initially visible to the robot's on-board camera, but its presence must be inferred from other visual cues. We present a novel detection algorithm which exploits scene texture [11] to infer candidate stair locations from a far distance, and optical flow [1] to precisely localize the leading-stair edge during the near approach. This in turn enables accurate stair alignment, and traversal. We have validated our approach in real-world experiments, and demonstrate the performance of the proposed algorithm in practice. In what follows, we present an in-depth description of our robust stair-descending procedure.

III. Algorithm Overview

We denote the robot's initial frame of reference as $\{R_0\}$. As the robot travels, the robot-affixed frame $\{R_t\}$ changes in time. The three-d.o.f. orientation of the robot at time t is described by the quaternion of rotation ${}^{R_0}_{R_t}\mathbf{q}$, which we estimate with a 3D-Attitude EKF. From the quaternion, we extract the pitch α and yaw θ components of the robot's orientation.

A. Data-Flow Description

1) Stair perception: As depicted in Fig. 2(left), the stair perception module takes the camera data as input, and performs texture analysis, optical flow, and line extraction on the images. The lines are passed to the estimation module, to be utilized for orientation updates. The heading reference θ_r is passed to the controller to guide the robot during different phases of the algorithm [see Fig. 2(right)].

2) Estimation: The estimation module receives measurements from the tri-axial gyroscope in order to compute the current attitude of the robot, with respect to the initial frame $\{R_0\}$. This information is fused with heading updates based on the detected stair edges in a 3D-Attitude EKF in order to obtain high-accuracy estimates. The estimation module provides the controller with estimates of θ, $\dot{\theta}$, and α.

3) Controller: The controller takes as input the current estimates of θ, $\dot{\theta}$, and α from the estimator, as well as a heading reference signal θ_r, which is the desired heading direction. This quantity will change according to which phase the robot is in (e.g., driving towards a candidate stair location or aligning to the stairs).

B. Algorithm-Flow Description

1) Far-approach: During the far-approach phase (P1), the robot uses a texture energy measure to generate hypotheses (image regions) for possible descending-stair locations, and selects one to investigate from a closer distance. As the robot moves, it tracks the candidate stair location, and the reference heading direction θ_r is set to coincide with the unit-vector towards the centroid of the tracked region. When the tracked region becomes sufficiently large in the image, we transition to the near-approach phase (P2).

2) Near-approach: In the near-approach phase, the robot verifies whether the current descending-stair-location candidate is valid or not. To this end, we exploit optical flow and image line features in order to identify the depth discontinuity at the leading stair edge. If the stair edge is detected, the robot transitions to the alignment phase, otherwise, it transitions back to the far-approach phase to investigate another hypothesis for the stair location.

3) Alignment: During the alignment phase, the robot tracks the leading-stair edge detected during the near approach and uses the edge direction to compute θ_r and align perpendicular to the stairs.

4) Stair Traversal: After the alignment phase the robot traverses the stairs. During this maneuver, the robot maintains a safe distance from both the left and right staircase boundaries. When the robot is in the center of the stairs, the reference heading is perpendicular to the stair edges. When the robot is near the left or right boundaries of the staircase, θ_r is selected to steer the robot back towards the center.

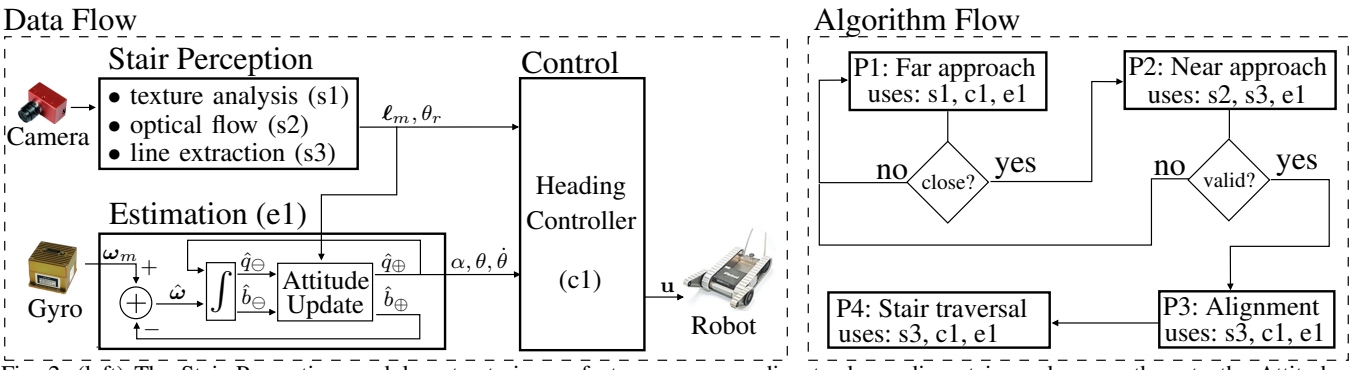

Fig. 2: (left) The Stair Perception module extracts image features corresponding to descending stairs and passes them to the Attitude-Estimation and Control modules. The Attitude-Estimation module fuses rotational velocity measurements from the gyroscope, with observations of straight-line edges extracted from the camera images, in order to estimate the three-d.o.f. orientation of the robot. The Control module drives the heading and velocity of the robot to follow safe trajectories. (right) The Algorithm flow comprises four phases: P1 uses texture analysis to generate hypotheses for possible stair locations. P2 uses line extraction and optical flow to determine the stair location and the boundary of the first step precisely. P3 aligns the robot to the stairs. P4 drives the robot down the stairs.

IV. Stair Perception

A. Far-approach stair detection

In this section, we present our vision-based approach for descending-stair detection. Fig. 3(a) shows a typical image of an indoor environment observed by a mobile robot equipped with a monocular gray-scale camera. Descending-stair detection is challenging in this case since the stairs are not directly visible in the image, and cannot be extracted by means of image features such as points or edges. Thus our algorithm must infer the presence of the descending stairs by exploiting other visual cues.

Humans can detect possible descending-stair areas by perceiving the relative change in depth of surrounding scene elements. To do this, they utilize a wide set of visual cues (e.g., monocular, stereo, motion parallax, and focus) [12]. Among these, we are primarily interested in monocular information, since the other visual cues require larger motions of the camera, as the scene depth increases. However, inferring the relative depth from a *single* image is difficult, because depth typically remains ambiguous given only local features. Thus, it is imperative to use a strategy that takes into account the overall structure of the image.

In our proposed solution, we exploit monocular cues, in particular texture variations, since they can be extremely useful for assessing changes in depth [13] (e.g., a carpet will exhibit a different texture resolution when observed at different distances). Among the existing texture descriptors, some of the most powerful ones use *texture energy* measures [14], which encode the amount of texture variation within a filtered window around a pixel. In [15], a supervised-learning approach was proposed to estimate the 3D depth from a single image. However, it relies on color-image processing and suffers from large computational burden. Instead, in our work we adopt Laws' texture-energy measures [11], due to their efficiency and computational speed.

In the first texture-transformation step, the current image $\mathcal{I}$ [see Fig. 3(a)] is filtered with a set of 5×5 masks $\mathcal{M}_k$,

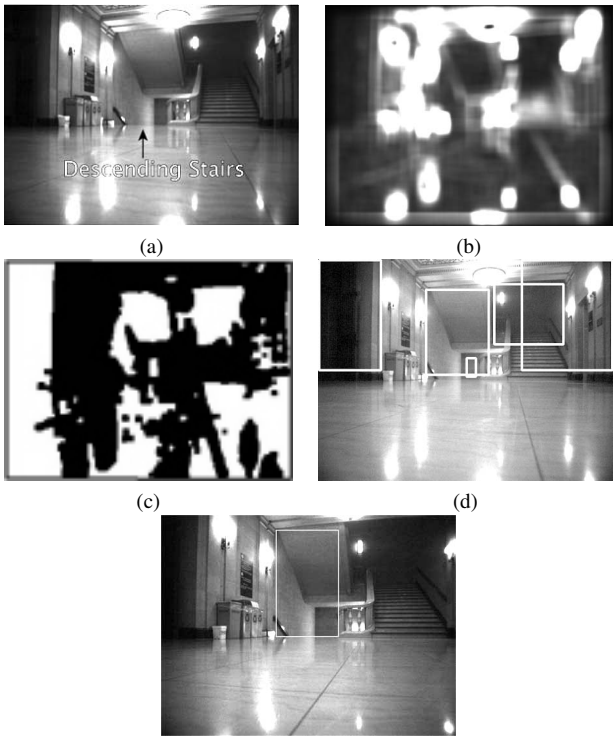

Fig. 3: Far-approach phase: (a) An image of the indoor environment. (b) The texture-energy measure highlights candidate stair locations (low gray-level). (c) The binary image obtained after adaptive thresholding. (d) Candidate boxes indicate possible descending stair locations. (e) One candidate is selected and tracked over consecutive images.

$k = 1, ..., 16$:

$$L5^TE5,\ E5^TL5,\ L5^TR5,\ R5^TL5,\ E5^TS5,\ S5^TE5,$$
$$S5^TS5,\ L5^TL5,\ R5^TR5,\ E5^TE5,\ L5^TS5,\ S5^TL5,$$
$$E5^TR5,\ R5^TE5,\ S5^TR5,\ R5^TS5,$$

where the letters indicate Local averaging, as well as the sensitivity to Edges, Spots, and Ripples, for each of the four

5×1 basic masks:

$$E5 = [-1, -2, 0, 2, 1],\ S5 = [-1, 0, 2, 0, -1],$$
$$R5 = [-1, -4, 6, -4, 1],\ L5 = [1, 4, 6, 4, 1].$$

From the obtained filtered images $\mathcal{F}_k$, the second texture-transformation step computes the local magnitudes of these quantities. We apply a *local-energy* operator to each filtered image to produce the texture-energy images $\mathcal{E}_k$[1]

$$\mathcal{E}_k(l,m) = \sum_{i=l-p}^{l+p} \sum_{j=m-p}^{m+p} |\mathcal{F}_k(i,j)|, \qquad k = 1, ..., 16. \quad (1)$$

In our experiments, we selected a window of size $p = 7$ and combined the various energies as

$$\mathcal{E}(l,m) = \sum_{k=1}^{16} \mathcal{E}_k(l,m), \quad (2)$$

which provided satisfactory performance.

Fig. 3(b) shows the final energy image $\mathcal{E}$ which has high-energy values for pixels corresponding to close objects such as railings, trashcans, and ascending stairs. In contrast, far objects exhibit low-energy values, and hence indicate possible descending-stair locations.

After computing the energy image, in the third step we apply an adaptive threshold[2] to obtain a binary image in which the 1s correspond to low-energy regions [Fig. 3(c)]. From these binary regions a series of connected contours have been extracted and each one has been fitted with a bounding box. Among all these boxes, we discard those portions that lie below the horizontal line passing through the image principal point. Note that, in the case of planar robot motion, when far from the stairs, this line constitutes a good approximation of the horizon line, and thus, it upper-bounds the floor plane. The remaining regions constitute the set of possible descending-stair locations, as shown in Fig. 3(d).

At this point the algorithm randomly selects one of the candidates, computes the reference heading θ_r with respect to its centroid, and uses it as an input to the controller (see Sect. IV-E). The robot moves towards the centroid of the chosen box, and tracks it through consecutive images using a nearest-neighborhood approach [Fig. 3(e)].

B. Near-approach stair detection

As the robot approaches the selected candidate location, it needs to assess whether it is a descending staircase or not. If the robot was equipped with a 3D sensor (e.g., stereo camera or 3D LADAR), it would be able to directly observe the staircase and construct its 3D model, which would facilitate stair descending. However, our goal in this work is to address all exteroceptive-sensing needs with a monocular camera. To this end, we exploit multiple visual cues (i.e., optical flow and image lines), in order to verify the stair hypothesis, and identify the leading stair edge.

1) Line extraction: As the robot approaches the stairs [see Fig. 4(a)], it extracts all lines present in the image. The goal is to determine the line corresponding to the stair boundary, so the algorithm retains only image lines which satisfy the following requirements: (i) they should lie in the lower half of the image, (ii) they are not near vertical in the image plane, and (iii) their length exceeds a specified percentage of the image resolution. This line-selection approach generates several hypotheses for the leading stair edge [see Fig. 4(b)], however, because no depth estimates are available from the line-extraction process, it is impossible to determine the stair edge with this information alone.

2) Variational optical flow: In order to infer which line, if any, corresponds to the stair boundary, we combine the image lines extracted in the previous step, with dense optical flow computed between sequential images. Traditional optical flow methods rely on correlating points or patches across two images in order to determine the scene motion. However, we have observed numerous cases in which correlation-based optical flow produces inaccurate results during the near-approach phase, due to lack of sufficient texture on the floor and walls. For this reason, we have employed a variational optical flow method, introduced by Brox et al. [1], which imposes additional constraints in the flow field to ensure consistency in low-texture regions [see Fig. 4(c)].

Specifically, for each candidate line extracted in an image, we compute the magnitude of the median flow within a small window above the line, ρ_A, and below the line, ρ_B. Since the line corresponding to the true stair edge will have large optical flow below the line [denoted by the brightly colored region, ρ_A, in Fig. 4(c)], and low optical flow above the line [denoted by the dark region, ρ_B, in Fig. 4(c)], we utilize a ratio test to determine which lines may correspond to the leading stair edge (i.e., $\rho_B/\rho_A \geq \epsilon$), and keep the ones which pass this test.

3) Voting for a candidate line: Typically, up to two or three candidate lines may be selected from the previous step. Hence, we employ a final step, which is a simple voting scheme, to detect if one of the candidate lines is the leading stair edge. Specifically, we cast three votes: (i) one for the longest length line, (ii) one for the line with the largest ratio ρ_B/ρ_A, and (iii) one for the line which is highest in the image. If a line receives two or three votes, it is selected as the leading stair edge [see Fig. 4(d)], otherwise the algorithm declares that no staircase is present, and returns to the far-approach phase as described in Fig. 2.

C. Stair-alignment phase

After the robot completes the near-approach phase and verifies that an open stairwell is present, it must rotate to align with the staircase. This is achieved by computing the unit vector along the stair-edge direction with respect to the initial robot frame $\{R_0\}$, and subsequently, calculating the reference heading direction θ_r, allowing the robot to determine the appropriate control inputs (see Sect. IV-E).

[1]Although [11] used both squared and absolute magnitudes to estimate the texture energy, we only utilize absolute magnitude, since it requires less computation and gives comparable performance in practice.

[2]While the initial value of this threshold is selected manually, we designed an adaptive thresholding strategy for coping with illumination changes. More details about this process are provided in Sect. V.

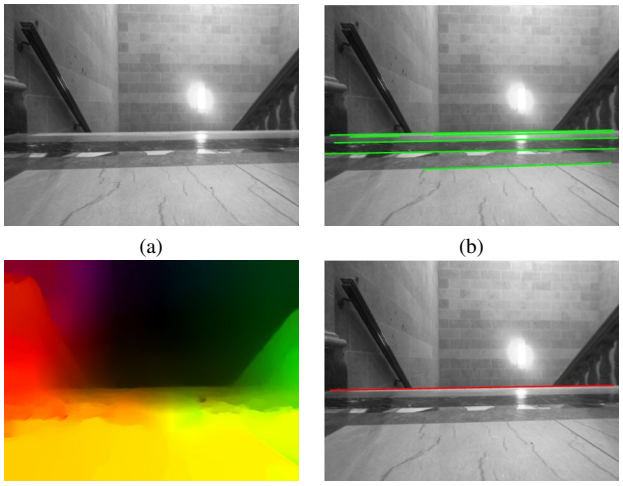

Fig. 4: Near-approach phase: (a) Image recorded during the near-approach. (b) Multiple lines are extracted which could correspond to the leading stair edge. (c) Optical flow is computed between consecutive images and correlated with the detected lines. The color indicates flow direction, and the brightness indicates flow magnitude. (d) The leading stair edge is determined and marked in red.

1) Determining the direction of the stair edge: Figure 5 depicts the observation of the leading stair edge. Each line projected onto the image plane is described by

$$ {}^{R_t}\boldsymbol{\ell}_j = \begin{bmatrix} \cos\phi_j & \sin\phi_j & -\rho_j \end{bmatrix}^T, \tag{3} $$

where $\{\phi_j, \rho_j\}$ are the line's polar coordinates in the current camera frame of reference. Thus, any homogeneous image point $\mathbf{p} = \begin{bmatrix} u & v & 1 \end{bmatrix}^T$ lies on the line iff $\mathbf{p}^T {}^{R_t}\boldsymbol{\ell}_j = 0$. We denote the plane which contains the projected line and the camera origin by Π_j, and remark that the normal vector to Π_j is ${}^{R_t}\boldsymbol{\ell}_j$. Furthermore, ${}^{R_t}\boldsymbol{\ell}_j$ is perpendicular to the stair-edge direction, since the stair edge lies in Π_j.

We note that, since the stair edge lies in a plane parallel to the yz-plane of $\{R_0\}$, the vector ${}^{R_0}\mathbf{e}_1 = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}^T$ is also perpendicular to the stair edge. By using the estimated orientation of the robot with respect to the global frame (see Sect. IV-E), we express the measured line-direction in the initial robot frame $\{R_0\}$, and compute the unit vector along the stair edge as

$$ {}^{R_0}\mathbf{s}_i = {}^{R_0}\mathbf{e}_1 \times {}^{R_0}_{R_t}\mathbf{C}\, {}^{R_t}\boldsymbol{\ell}_j. \tag{4} $$

To obtain a better estimate of the stair-edge direction, we compute its average over several observations as the robot aligns to the stair.

2) Computing the desired heading direction: Let the unit vector along the stair edge be ${}^{R_0}\mathbf{s}_i \triangleq \begin{bmatrix} 0 & y_s & z_s \end{bmatrix}^T$. Then the reference heading of the robot, θ_r, is the angle which aligns the robot's z-axis with the perpendicular to ${}^{R_0}\mathbf{s}_i$, so that the robot is facing straight down the stairs. Thus, the unit-vector direction which the robot should head in is ${}^{R_0}\mathbf{v}_r = \begin{bmatrix} 0 & z_s & -y_s \end{bmatrix}^T$, and the reference heading is $\theta_r = \text{atan2}\,(z_s, -y_s)$.

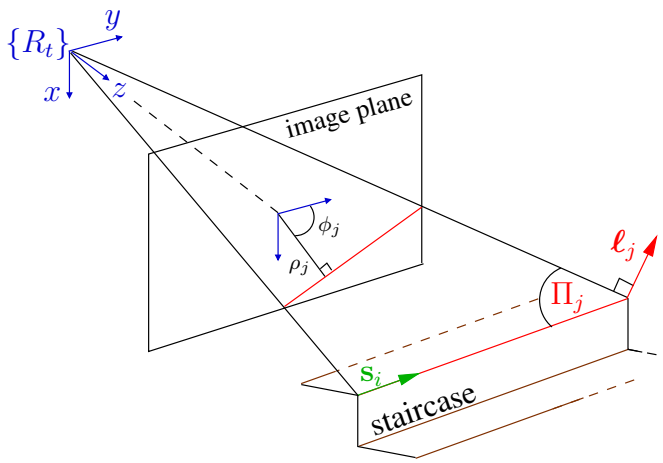

Fig. 5: Camera observation of a stair edge. The camera observes the projection of the stair edge onto the image plane, as a line with parameters ρ_j and ϕ_j. The plane passing through the focal point and the projected line in the image plane is Π_j, whose normal vector is $\boldsymbol{\ell}_j$. The stair direction $\mathbf{s}_i$ lies in Π_j and is perpendicular to $\boldsymbol{\ell}_j$.

D. Stair traversal

During the stair traversal phase, the robot must drive onto the plane of the stairs, traverse the staircase going down, and then drive off the stairs. Initially the robot is aligned perpendicular to the stair-edge direction, and positioned roughly in the center of the stairwell opening. The robot commands a small linear velocity to move forward until its center of gravity passes over the first stair edge and it settles onto the stairs. We detect the floor-to-stair transition based on the robot's pitch estimate (see Sect. IV-E).

While the robot is on the staircase, the reference heading direction fed to the controller depends on the location of the robot on the stairs. We employ a safe-center strategy, as in [6], which defines three stair regions. On the left and right sides of the stairs, there is a no-drive region, where the robot is at risk of colliding with the wall or railing. The middle of the stair-case represents a safe-to-drive region where the robot should pass through.

We continuously monitor the location of the robot on the stairs using a ratio of left and right end points of the detected stair edges in the image. If the robot is in the center of the stairs, then the reference heading direction is the perpendicular stair direction (i.e., the robot drives straight down the stairs). If the robot experiences slippage, it may end up in the left or right no-drive regions. In this case, the reference heading direction changes, to steer the robot back to the middle of the stairs. As a last step, the robot detects the transition from the bottom stair onto the floor using a threshold of the estimated robot's pitch estimate, and automatically stops.

E. Attitude estimation and heading control

We employ an EKF for three-d.o.f. attitude estimation that fuses gyroscope measurements with orientation updates from the observed stair edges. The filter state is $\mathbf{x} = \begin{bmatrix} \mathbf{q}^T & \mathbf{b}^T \end{bmatrix}^T$, where $\mathbf{b}$ denotes the time-varying biases that affect the

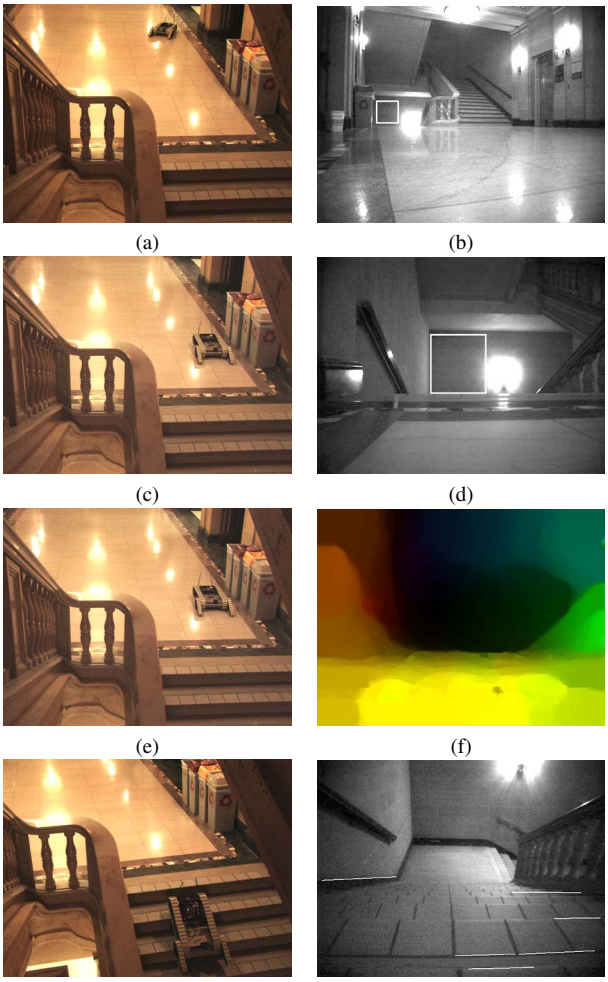

Fig. 6: Experiment 1. Stair detection, approach and traversal; (a) The robot in phase 1 steers towards the stair hypothesis which is boxed in white, and depicted in (b). (c) The robot correctly tracks the hypothesis for the stair location up to the stairwell opening (d), and transitions to phase 2. (e) During near-approach, the robot executes a small translational motion in order to compute the optical flow field (f), and precisely locates the leading stair edge. (g) The robot drives onto the stairway, and tracks lines as it traverses down (h).

gyroscope measurements, and $\mathbf{q}$ denotes the robot's attittude represented by the quaternion of rotation. We employ a PID controller for the robot heading, derived based on the linearized system model, which generates velocity commands as a function of the estimated orientation of the robot $\hat{\mathbf{q}}$, and the desired heading direction θ_r. Details for the attitude estimation and heading control are provided in [6].

V. Experimental Validation

A. Hardware and software description

The proposed method was developed and implemented on an iRobot Packbot. The Packbot is a tracked, skid-steered robot of approximately 66 cm in length and 23 kg in weight. The robot is equipped with an on-board navigation computer (PC-104 stack), as well as a tri-axial gyroscope (ISIS IMU) which measures the robot's three-d.o.f. rotational velocity at 100 Hz, and a monocular grey-scale camera (Videre Design) which records 640×480 px images at 30 Hz. The sensors' intrinsic parameters (i.e., gyro noise parameters, as well as camera focal length, optical center, skew coefficients, and radial distortion parameters) are computed off-line. A precise estimate of the camera-to-IMU transformation has also been computed using the approach presented in [16].

The proposed descending-stair navigation method was implemented in C++ under the GNU/Linux operating system. The texture analysis and line extraction has been developed using Intel's OpenCV computer vision library. The variational optical flow [1] was computed remotely on a server, due to the limited computational resources available on the Packbot. The remote transfer and processing step takes approximately 10 seconds to complete, and must be performed only once per trial (during the near-approach phase). All other implemented algorithms run on the robot in real-time, at rates between 15 Hz and 30 Hz.

B. Experimental setup and results

We evaluated the proposed approach under typical lighting conditions on a stair-case at the University of Minnesota, and we hereafter present the most representative results from two tests.[3]

The robot is typically positioned 10 to 15 m away from the descending-stair portion of the environment. In order to compensate for commonly occurring changes in illumination, we implemented a method for adaptively tuning the threshold value used for converting the texture-energy image to the binary one (see Sect. IV-A). Our policy increases (decreases) the value of the threshold when the area of the tracked region shrinks (grows) over a specific percentage between two consecutive frames. This strategy provided satisfactory performance of the region tracking, even in the presence of illumination changes and across large distances.

Fig. 6 shows the experimental results in the case that the tracked region corresponds to the descending stairs; Fig. 6(a)-(b) presents the phase 1 portion of the experiment, in which the robot is in the initial position and one possible descending-stair region is selected from the texture analysis. After phase 1 is completed, the robot is near the edge of the descending stairs [Fig. 6(c)] and the candidate stair location has been successfully tracked over time [Fig. 6(d)]. At this point, the robot transitions into phase 2, and performs a small translational motion in order to compute the optical-flow field between subsequent images and detect the leading stair edge [Fig. 6(e)-(f)]. Once the stair edge is correctly detected, the robot aligns to it and begins stair descent [Fig. 6(g)-(h)].

We have tested our strategy also in the case that the initial region selection does not coincide with an actual descending stair (in this experiment the robot aims towards a doorway). Fig. 7(a)-(b) shows the initial robot position and the selection of the doorway from the texture analysis. This region is tracked during the entire phase 1 [Fig. 7(c)-(d)]. However, during the execution of phase 2 [Fig. 7(e)]

[3] A video documenting the presented experiment, as well as an additional experiment on a stair-case with different texture, material, and lighting conditions is available at http://mars.cs.umn.edu/videos/StairDescend.mp4

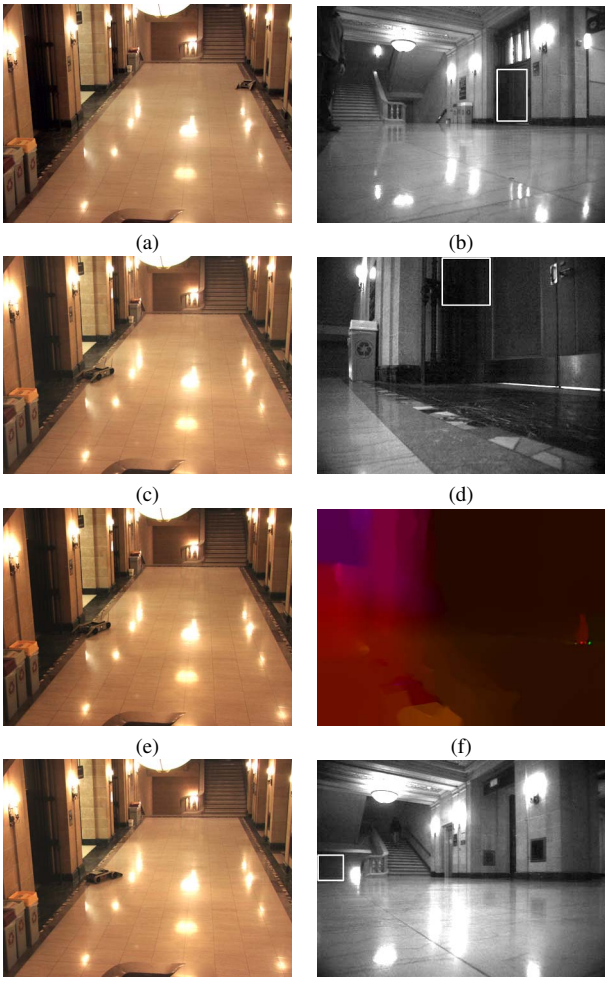

Fig. 7: Experiment 2. Detecting a wrong descending-stair location; (a) The robot in phase 1 steers towards a wrong hypothesis (i.e., a doorway) which is boxed in white, and depicted in (b). (c) The robot arrives near by the region and (d) tracks it correctly during phase 1. (e) In the near-approach, the robot executes a small translational motion in order to compute the optical flow field (f), which does not exhibit any discontinuity across prominent image lines. (g) The robot rotates and (h) computes a new candidate for the descending stair location.

the robot rejects this candidate stairwell, since the flow field does not exhibit a depth discontinuity across any prominent image lines [Fig. 7(f)]. At this point the robot rotates $180°$, and begins searching for a new candidate region [Fig. 7(g)-(h)].

VI. Conclusions and Future Work

In this paper, we presented a method for descending-staircase detection, approach, and traversal using inertial sensing and a monocular camera. We developed a four-stage strategy to address this problem, which includes far-approach, near-approach, stair alignment, and stair traversal phases. In each step we exploited salient stair features in the image in order to improve robustness. One of the key challenges we overcame was detecting regions of interest (i.e., stairwells) in an image, which essentially manifest themselves as gaps. To this end, we described and implemented two methods, one for far-approach, which relies on texture energy measures, and one for near-approach, which exploits optical flow discontinuities to determine the leading stair edge. Making our algorithm robust to initial-illumination conditions is part of future work.

In our ongoing work, we are investigating methods for on-line learning [i.e., kernel-based support vector machines (SVM)], which will enable the robot to detect new staircases based on previous experience with other stairs. Additionally, we plan to exploit information for the presence of ascending stairs [2] to assign probabilities to the different hypotheses for the location of descending stairs. Finally, we intend to extend the results of our work to the case of humanoid robots whose cameras move outside the plane as the robot walks, and whose viewing directions can be controlled to increase the acquired information.

References

[1] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert, "High accuracy optical flow estimation based on a theory for warping," in *Proc. of the European Conf. on Computer Vision*, Prague, Czech Republic, May 11–14, 2004, pp. 25–36.

[2] N. Molton, S. Se, M. Brady, D. Lee, and P. Probert, "Robotic sensing for the partially sighted," *Robotics and Autonomous Systems*, vol. 26, no. 2-3, pp. 185–201, Feb. 1999.

[3] J. D. Martens and W. S. Newman, "Stabilization of a mobile robot climbing stairs," in *Proc. of the IEEE Int. Conf. on Robotics and Automation*, San Diego, CA, May 8–13, 1994, pp. 2501–2507.

[4] G. Chen, M. Xie, Z. Xia, L. Sun, J. Ji, Z. Du, and W. Lei, "Fast and accurate humanoid robot navigation guided by stereovision," in *Proc. of the IEEE Int. Conf. on Mechatronics and Automation*, Changchun, China, Aug. 9–12, 2009, pp. 1910–1915.

[5] G. Figliolini and M. Ceccarelli, "Climbing stairs with EP-WAR2 biped robot," in *Proc. of the IEEE Int. Conf. on Robotics and Automation*, Seoul, Korea, May 21–26, 2001, pp. 4116–4121.

[6] A. I. Mourikis, N. Trawny, S. I. Roumeliotis, D. M. Helmick, and L. H. Matthies, "Autonomous stair climbing for tracked vehicles," *Int. Journal of Robotics Research*, vol. 26, no. 7, pp. 737–758, Jul. 2007.

[7] Q.-H. Vu, B.-S. Kim, and J.-B. Song, "Autonomous stair climbing algorithm for a small four-tracked robot," in *Proc. of the IEEE Int. Conf. on Control, Automation, and Systems*, Seoul, Korea, Oct. 14–17, 2008, pp. 2356–2360.

[8] S. Steplight, G. Egnal, S.-H. Jung, D. B. Walker, C. J. Taylor, and J. P. Ostrowski, "A mode-based sensor fusion approach to robotic stair-climbing," in *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, Takamatsu, Japan, Oct. 31–Nov. 5, 2000, pp. 1113–1118.

[9] Y. Xiong and L. Matthies, "Vision-guided autonomous stair climbing," in *Proc. of the IEEE Int. Conf. on Robotics and Automation*, San Francisco, CA, Apr. 24–28, 2000, pp. 1842–1847.

[10] D. M. Helmick, S. I. Roumeliotis, M. C. McHenry, and L. H. Matthies, "Multi-sensor, high speed autonomous stair climbing," in *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, Lausanne, Switzerland, Sep. 30–Oct. 5, 2002, pp. 733–742.

[11] K. Laws, "Rapid texture identification," in *Proc. of the SPIE Conf. on Image Processing for Missile Giudance*, no. 238, San Diego, CA, Jul. 28–Aug. 1, 1980, pp. 376–380.

[12] J. Loomis, "Looking down is looking up," *Nature News and Views*, vol. 414, pp. 155–156, 2001.

[13] J. Malik and P. Perona, "Preattentive texture discrimination with early vision mechanisms," *Journal of the Optical Society of America A*, vol. 48, no. 2, pp. 75–90, 1990.

[14] E. Davies, *Machine Vision: Theory, Algorithms, Practicalities*, M. Kaufman, Ed. Elsevier, 2005.

[15] A. Saxena, S. Chung, and Y. A. Ng, "3D depth reconstruction from a single still image," *Int. Journal of Computer Vision*, vol. 76, no. 1, pp. 53–69, Jan. 2008.

[16] F. M. Mirzaei and S. I. Roumeliotis, "A Kalman filter-based algorithm for IMU-camera calibration: Observability analysis and performance evaluation," *IEEE Trans. on Robotics*, vol. 24, no. 5, pp. 1143–1156, Oct. 2008.