Autonomous Stair Climbing for Tracked Vehicles

Anastasios I. Mourikis[1], Nikolas Trawny[1], Stergios I. Roumeliotis[1], Daniel M. Helmick[2], and Larry Matthies[2]

***Abstract*— In this paper, we present an algorithm for autonomous stair climbing with a tracked vehicle. The proposed method achieves robust performance under real-world conditions, without assuming prior knowledge of the stair geometry, the dynamics of the vehicle's interaction with the stair surface, or lighting conditions. Our approach relies on fast and accurate estimation of the robot's heading and its position relative to the stair boundaries. An extended Kalman filter is used for quaternion-based attitude estimation, fusing rotational velocity measurements from a 3-axial gyroscope, and measurements of the stair edges acquired with an onboard camera. A two-tiered controller, comprised of a centering- and a heading-control module, utilizes the estimates to guide the robot fast, safely, and accurately upstairs. Both the theoretical analysis and implementation of the algorithm are presented in detail, and extensive experimental results demonstrating the algorithm's performance are described.**

***Index Terms*— Stair Climbing, Autonomous Robots, Inertial Sensing, Attitude Estimation, Computer Vision.**

I. Introduction

STAIRWAYS and steps are omnipresent in man-made environments. Designed to easily bridge large vertical distances for humans, stairs represent a serious challenge to vehicles and robots. In order for robots to operate efficiently in urban environments, this challenge needs to be addressed. Robotic stair climbing can be applied in numerous scenarios, for example, in urban search and rescue missions, in military operations, to increase mobility of handicapped people, or to improve the efficiency of household helping robots. For these reasons, *autonomous* robotic stair climbing has been the subject of ongoing research in the last years.

In many current applications, mobile robots are still teleoperated, with only limited autonomy. Climbing stairs, as for example required in search and rescue missions in urban areas, is very demanding on a human operator [1]. Usually the robot maneuvers outside the field of view of the operators, forcing them to rely only on feedback from the robot's camera. The latter is usually mounted very close to the ground, has a narrow field of view, and the returned images are often blurred due to the robot's highly dynamic motion. This greatly impairs the operator's perception of the vehicle's current spatial orientation. Combined with the latency in data transmission and the robot's high slippage on the stair edges, this can result in inaccurate and slow stair climbing, collisions with the stair walls, and even in toppling of the vehicle. It is therefore desirable to endow a robot with *autonomous stair-climbing capabilities*, thus enabling faster, safer, and more precise operation while at the same time reducing the user load.

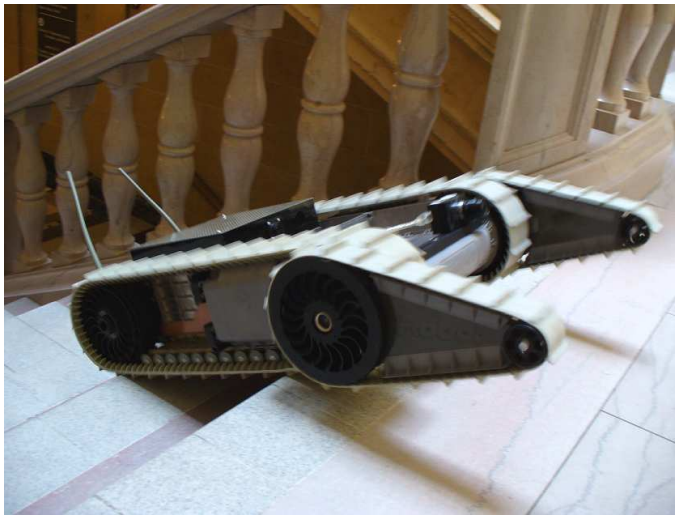


Fig. 1. Robot climbing stairs autonomously. Picture taken at the Digital Technology Center, University of Minnesota.

The controller employed for autonomous stair climbing requires frequent and precise estimates of the vehicle's position and heading relative to the staircase, in order to safely guide it up the stairs. The motion profile (high slippage, shocks) and the complex interactions of the robot tracks with the stair render exact modeling of the vehicle-ground dynamics intractable. Besides, an overly detailed model would prohibit the algorithm from being flexible and robust over a wide range of parameter values, such as stair dimensions and surface material. At the same time, the number of required sensors on the robot should be kept as low as possible in order to minimize cost, weight, and power consumption. In order to maximize speed and reduce the risk of collision or toppling, it is necessary to maintain the robot heading approximately perpendicular to the stair edges. This can be accomplished by a heading controller based solely on vehicle dynamics, if combined with an accurate, high-bandwidth attitude estimator.

In this paper, we outline an algorithm that allows robust, safe, fast, and accurate traversal of stairs of various dimensions, using a 3-axial gyroscope and a single camera as the only sensors. An extended Kalman filter (EKF) integrates the angular velocities measured by the gyroscopes to form an orientation estimate. This estimate is then updated using measurements of the projections of stair edges, extracted from the camera images. Furthermore, the stair edge observations allow estimating the robot's offset relative to the center of the staircase. These values are used by a two-tiered controller

[1] University of Minnesota, Department of Computer Science and Engineering, Minneapolis, MN, {mourikis, trawny, stergios}@cs.umn.edu

[2] Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA, dhelmick@robotics.jpl.nasa.gov, lhm@telerobotics.jpl.nasa.gov

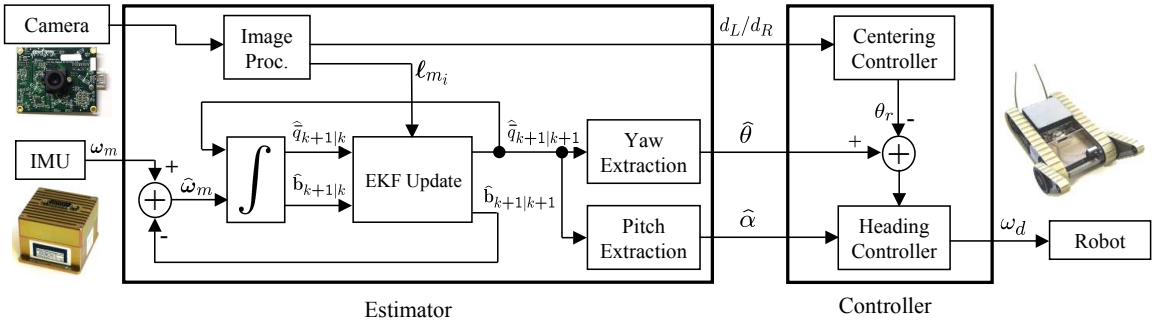


Fig. 2. The block diagram of the stair-climbing algorithm.

(cf. Fig 2) to guide the vehicle upstairs. This algorithm is very versatile and can be applied, for example, on an iRobot PackBot [2], as the one used in our experiments (cf. Section V), on a Remotec Andros remote vehicle [3], or on any tracked robot equipped with gyroscopes and a camera.

The remainder of this paper is structured as follows. After an overview of related work in Section II, we present our algorithms for estimating both the robot's attitude as well as its deviation from the center of the stairs in Section III. Our proposed control algorithm is outlined in Section IV. We have successfully implemented and tested these algorithms on a tracked robot, for which we present experimental results in Section V.

II. Related Work

Stair climbing has been carried out with robots using different types of locomotion. One can roughly distinguish wheeled, legged, and tracked robots.

A. Wheeled Robots

Wheeled robots usually have to resort to mechanic extensions to overcome stairs. One application of such a technique is in patient rehabilitation, where stair climbing could greatly enhance mobility, and thus quality of life, of people confined to wheelchairs. Lawn and Ishimatsu [4] present a stair-climbing wheelchair using two (forward and rear) articulated wheel clusters attached to movable appendages. The robot is equipped with step-contact sensors, but relies on user steering and is thus only *semi-autonomous*.

B. Legged Robots

In [5], Figliolini and Ceccarelli present the architecture of the bipedal robot EP-WAR2, that uses electropneumatic actuators and suction cups for locomotion. In order to climb stairs, the robot relies on an *open-loop* control algorithm implemented as a finite-state machine. The main limitation of the approach is that operating in a different staircase necessitates manual recalibration.

Albert *et al.* [6] implemented a stair-climbing algorithm on the bipedal robot BARt-UH. The authors employ *stereo vision* and the projection of a laser line in order to estimate stair dimensions. These are then used in a planning algorithm that produces piecewise analytical joint trajectories. The trajectory parameters are tabulated for different stair dimensions and interpolated as needed. Therefore, BARt-UH can be considered an autonomous stair climber. However, the demanding control of a legged robot, due to its higher center of gravity and its intricate actuation, result in high computational load and overall system complexity. This severely limits the robot's speed during stair climbing.

The humanoid robots of Sony and Honda, QRIO and ASIMO, are also capable of autonomous stair climbing. QRIO [7] employs *stereo vision* to segment planar surfaces. These surfaces are used in a path planning algorithm that allows the robot to climb up and down stairs, sills and ledges. In [8], Hirai *et al.* outline the foot placement algorithm employed in Honda's humanoid ASIMO. Both robots use dense stereo vision, requiring the robots to move slowly in order to ensure image quality.

Stair climbing with a hexapod robot has been demonstrated by Moore *et al.* [9]. The robot RHex makes use of a special curved leg design and pre-programmed leg trajectories, rendering it capable to climb stairs of various dimensions. The employed algorithm, however, is strictly *open-loop*. It is thus unable to prevent collisions with the stair walls or balustrades, and cannot compensate large heading deviations induced by slippage or shocks.

C. Hybrid Locomotion

Matsumoto *et al.* [10] have devised a hybrid biped leg-wheeled system, combining the advantages of wheeled locomotion with the greater flexibility of legs. They derive a wheel torque control algorithm to robustly position the robot's center of gravity, using gyroscopes, accelerometers, encoders, and torque/force sensors for feedback. The robot forward-tilt angle is estimated by a combination of angular velocity integration and gravity vector measurements, although details about the estimation of the center of gravity location are omitted. The torque derivations are based on a quasi-static analysis, assuming low robot speed and smooth motion. Moreover, the stair dimensions are used as parameters of the control law, but are not estimated online and therefore need to be known a priori.

D. Tracked Robots

Several works have examined stair climbing for tracked robots, which is within the focus of this paper. Tracked robots have a larger ground contact surface than wheeled vehicles,

and are more stable than bipeds due to their low center of gravity. Liu *et al.* [11] derive the fundamental dynamics of the stair-climbing process for a tracked robotic element, analyzing the different phases of riser climbing, nose crossing, nose line climbing and the effects of grouser bars or cleats. The analysis is limited to 2D, and slippage, shocks, and intermittent loss of track-surface contact, phenomena that are commonly encountered during stair climbing, are neglected. The resulting model is therefore not sufficiently accurate to allow exact trajectory prediction, but is well-suited for preliminary design studies of one- and multi-element tracked robots.

In [1], Martens and Newman note the difficulties involved in teleoperated stair climbing of tracked robots. This task is very demanding on the operator, due to limited sensor feedback and track slippage. The results are slow speed and inaccurate heading, which can lead to toppling of the robot. In order to allow semi-autonomous stair climbing, they develop a stabilizing feedback controller that enables the robot to maintain its heading, using only accelerometers. However, the fact that accelerometers measure both gravity and body accelerations can lead to large errors when employing these sensors to estimate the robot's attitude.

Steplight *et al.* [12] rely on measurements from sonar, a monocular camera, and two accelerometers for attitude estimation. The authors argue that these sensors are complementary, each providing reliable estimates under different conditions. An example is the above-mentioned use of accelerometer measurements to infer attitude using the gravity vector: this provides quite accurate results when the robot is standing still, but fails when the robot is subject to shocks and bumps. A so-called "broker module" determines which estimate to use at every time instant, depending on a confidence measure provided by each sensor. This confidence measure is largely based on heuristics, and is often inversely proportional to the deviation from the prior attitude estimates.

An approach for determining the robot's heading using *only* monocular vision is presented by Xiong and Matthies [13]. The algorithm extracts lines from stair images in order to determine the two quantities necessary for steering control, namely (i) the offset angle describing the robot heading relative to the stairs, and (ii) the ratio of the distances to the left and right boundaries of the staircase, which is an indicator for the relative distance from the centerline. This work is extended in [14], where an EKF is used to fuse 3D attitude information from gyroscope measurements, vision, and a laser scanner. The high frequency of the inertial measurements, and thus of the EKF, allows for high-bandwidth control that increases the robustness and accuracy of stair climbing significantly.

One of the main drawbacks of both [13] and [14] are the ad-hoc assumptions underlying the computation of the yaw estimate and its variance from the images. First, it is assumed that the robot is oriented parallel to the plane of the stair edges at all times (i.e., zero roll and constant pitch). This does not reflect the pronounced disturbances induced by slippage and bouncing (cf. the roll and pitch angle profiles during stair climbing shown in Fig. 3). Second, when the projections of the stair edges on the image plane are processed to estimate the yaw, its covariance is approximated by the inverse of

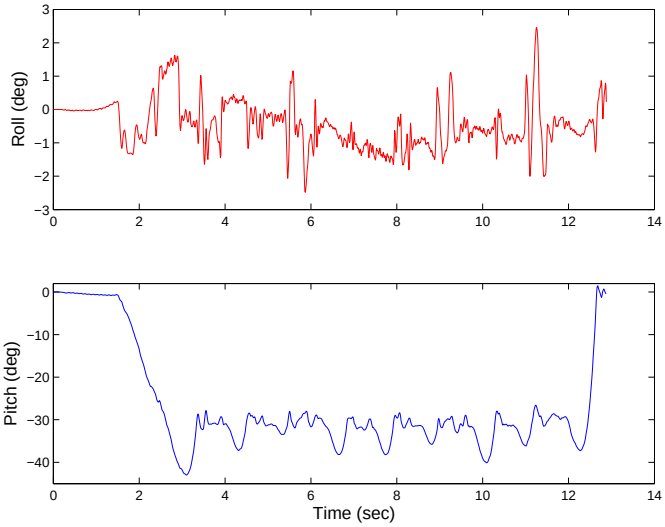


Fig. 3. The time-evolution of the robot's roll and pitch angles during a typical ascent. Note the significant variation in the pitch angle.

the squared y-intercept of the line on the image plane. This approximate yaw measurement and its associated variance is then provided to the EKF as an inferred measurement in order to update the attitude estimates. However, the imprecise approximations of both the derived yaw and its variance degrade the resulting attitude estimates.

This paper further improves the work presented in [14], in that a new measurement model is derived that allows *tight integration* of the visual information (that is, the detected stair edges) into the EKF, thus increasing robustness and accuracy of the attitude estimate. Additionally, an improved method to detect the ratio of the distances to the left and right wall is presented, based only on camera and gyroscope data. This is used for maintaining the robot's trajectory along the stair center, thus decreasing the risk of collision with the balustrades. Employing a camera for updating the attitude estimates, and keeping the robot close to the centerline, eliminates the need for a laser scanner, resulting in reduced mass, volume, cost, and power consumption. The presented analysis shows that our measurement model allows observability of two degrees of freedom when only stair edges (parallel to one global unit vector) are detected. Furthermore it is proven that the robot attitude becomes fully observable if at least one additional line (of known global direction, different from that of the stair edges) is detected in the image (cf. Appendix III).

III. Attitude and Distance Ratio Estimation

To safely control the robot's trajectory on the stairs, precise estimates of the robot's attitude, as well as of the distance ratio to the left and right boundaries (e.g., walls or railings) of the traversable surface of the stairs, are necessary. Due to the highly dynamic robot-surface interaction, resulting in significant slippage, odometry is not sufficiently accurate and reliable for this task. Instead, we employ an EKF to fuse rotational velocity measurements with measurements of the projections of stair edges on the camera images. In this

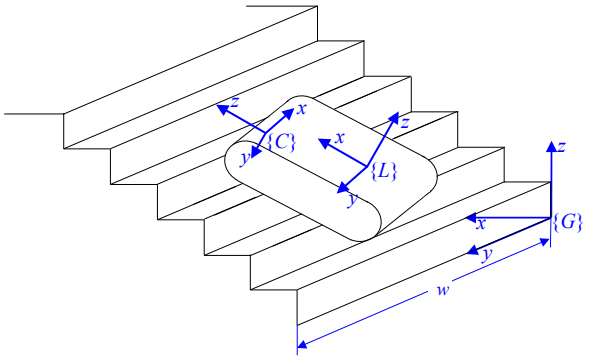


Fig. 4. The robot on the stairs with the defined frames shown: The global frame $\{G\}$ affixed to the stairs, the local frame $\{L\}$, attached to the robot, and the camera frame $\{C\}$. The width of the stairs is denoted by w.

section, we describe the various components of the estimation algorithm.

A. Attitude Estimation

1) Dynamic Model replacement: In order to estimate the robot's 3D attitude, it would be desirable to precisely model the robot dynamics, and treat the control commands as inputs. In our approach we employ sensor modeling instead, using the measurements from the gyroscopes to propagate the attitude estimate, and camera information to update it. The main reasons for this are: (i) dynamic modeling is dependent on robot and stair parameters, and would thus require calibration for every new stair, and (ii) dynamic model-based observers require a large number of states that increase the computational needs without producing superior results. This has been documented in the literature before; the interested reader is referred to [15], [16] and [17] for a detailed discussion.

2) Attitude Representation: The robot's attitude describes the relationship between the global coordinate frame $\{G\}$ and the robot-fixed local coordinate frame $\{L\}$. As shown in Fig. 4, $\{G\}$ is affixed to the stairs, such that the y-axis is parallel to the edges of the steps and the z-axis is pointing upwards. Additionally, we define a camera-fixed coordinate frame $\{C\}$, whose relationship to the local frame $\{L\}$ is known and constant.

The Euler angles yaw, pitch, and roll, which are the most commonly used attitude representation [18], are subject to singularities. The direction-cosine matrix, another popular representation, suffers from redundancy, comprising nine elements of which only three are independent. We have therefore selected the quaternion attitude representation, allowing for compact, singularity-free, and efficient attitude computation. The following derivations are largely based on [17], [19] and can be found in more detail in [20].

The four-element unit quaternion of rotation is defined as

$$\bar{q} = \begin{bmatrix} \mathbf{q} \\ q_4 \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{k}} \sin(\theta_q/2) \\ \cos(\theta_q/2) \end{bmatrix}, \qquad \bar{q}^{\mathrm{T}} \bar{q} = 1 \tag{1}$$

where $\hat{\mathbf{k}}$ is the unit vector along the axis of rotation, and θ_q denotes the rotation angle. Using the convention of [21], the product of quaternions is defined such that it corresponds to the product of rotation matrices in the same order, i.e.,

$${}^K_J\mathbf{C}({}^K_J\bar{q}) \cdot {}^J_I\mathbf{C}({}^J_I\bar{q}) = {}^K_I\mathbf{C}\left({}^K_J\bar{q} \otimes {}^J_I\bar{q}\right) \tag{2}$$

where ${}^J_I\mathbf{C}$ is the rotation matrix that expresses the basis vectors of frame $\{I\}$ in terms of frame $\{J\}$. We use the quaternion[1] $\bar{q} = {}^L_G\bar{q}$ to describe the global frame $\{G\}$ expressed in the local robot frame $\{L\}$. The correspondence between quaternion and rotation matrix is given by

$${}^L_G\mathbf{C}(\bar{q}) = \mathbf{I}_{3\times3} - 2q_4\lfloor\mathbf{q}\times\rfloor + 2\lfloor\mathbf{q}\times\rfloor^2 \tag{3}$$

where $\lfloor\mathbf{q}\times\rfloor$ denotes the skew-symmetric cross-product matrix

$$\lfloor\mathbf{q}\times\rfloor = \begin{bmatrix} 0 & -q_3 & q_2 \\ q_3 & 0 & -q_1 \\ -q_2 & q_1 & 0 \end{bmatrix} \tag{4}$$

Our controller uses as inputs the yaw and pitch angles, which can be extracted from the rotational matrix following the Euler X-Y-Z angles (roll-pitch-yaw) convention [22].

3) Attitude Kinematics: The time evolution of the quaternion depends on the rotational velocity $\boldsymbol{\omega}$ of the robot. Given $\boldsymbol{\omega}$, the attitude is governed by the differential equation

$$\dot{\bar{q}}(t) = \frac{1}{2}\,\boldsymbol{\Omega}(\boldsymbol{\omega}(t))\,\bar{q}(t) \tag{5}$$

where

$$\boldsymbol{\Omega}(\boldsymbol{\omega}(t)) = \begin{bmatrix} -\lfloor\boldsymbol{\omega}\times\rfloor & \boldsymbol{\omega} \\ -\boldsymbol{\omega}^{\mathrm{T}} & 0 \end{bmatrix} \tag{6}$$

In order to compute the attitude during robot operation, we employ a first-order numerical integrator [23] for the quaternion, assuming that $\boldsymbol{\omega}$ evolves linearly during the integration time step $\Delta t = t_{k+1} - t_k$. Under this assumption, we can integrate Eq. (5) as

$$\bar{q}_{k+1} = \left(\exp\left(\frac{1}{2}\,\boldsymbol{\Omega}(\boldsymbol{\omega}_a)\Delta t\right) + \frac{1}{48}\Big(\boldsymbol{\Omega}\big(\boldsymbol{\omega}_{k+1}\big)\boldsymbol{\Omega}\big(\boldsymbol{\omega}_k\big) - \boldsymbol{\Omega}\big(\boldsymbol{\omega}_k\big)\boldsymbol{\Omega}\big(\boldsymbol{\omega}_{k+1}\big)\Big)\Delta t^2\right)\bar{q}_k \tag{7}$$

where

$$\boldsymbol{\omega}_a = \frac{\boldsymbol{\omega}_{k+1} + \boldsymbol{\omega}_k}{2} \tag{8}$$

denotes the average rotational velocity during the integration interval $[t_k, t_{k+1}]$.

4) Gyroscope Sensor Model: Instead of the true rotational velocity required for the quaternion integration, the gyroscopes provide only a noise-corrupted measurement $\boldsymbol{\omega}_m$. The objective of the EKF is to obtain an estimate of the attitude by fusing these gyroscope measurements with additional information from a monocular camera. During propagation, the rotational velocity measurements $\boldsymbol{\omega}_m$ are integrated. The resulting estimate is corrected using stair-edge observations from the camera in the update step, which will be discussed in Section III-A.9.

In order to obtain the estimate, the EKF requires knowledge of the measurement noise characteristics. Noise in gyroscope

[1]For clarity of notation, we will henceforth drop the prescripts and simply denote the quaternion ${}^L_G\bar{q}$ representing the robot's attitude as $\bar{q}$.

measurements is known to be correlated [19]. We therefore employ a *noise shaping filter*, modeling the measured rotational velocity $\boldsymbol{\omega}_m$ as the true value $\boldsymbol{\omega}$ corrupted by the drift rate bias $\mathbf{b}$ and drift rate noise $\mathbf{n_r}$. The bias itself is modeled as a random walk process and included in the state vector, i.e., $\mathbf{x} = \begin{bmatrix}\bar{q}^T & \mathbf{b}^T\end{bmatrix}^T_{7\times1}$. The gyroscope measurement model can hence be written as

$$\boldsymbol{\omega}_m(t) = \boldsymbol{\omega}(t) + \mathbf{b}(t) + \mathbf{n_r}(t) \tag{9}$$

$$\dot{\mathbf{b}}(t) = \mathbf{n_w}(t) \tag{10}$$

where $\mathbf{n_r}, \mathbf{n_w}$ are independent, additive white Gaussian noise processes with zero mean

$$E[\mathbf{n_r}(t)] = \mathbf{0}, \quad E[\mathbf{n_r}(t)\mathbf{n_r}(t')^T] = \sigma_{r_c}^2\mathbf{I}_{3\times3}\delta(t-t') \tag{11}$$

$$E[\mathbf{n_w}(t)] = \mathbf{0}, \quad E[\mathbf{n_w}(t)\mathbf{n_w}(t')^T] = \sigma_{w_c}^2\mathbf{I}_{3\times3}\delta(t-t') \tag{12}$$

In the above expressions, $\delta(\cdot)$ denotes the Dirac delta function.

The state space model is

$$\begin{bmatrix}\dot{\bar{q}}\\ \dot{\mathbf{b}}\end{bmatrix} = \begin{bmatrix}\frac{1}{2}\,\boldsymbol{\Omega}(\boldsymbol{\omega}_m - \mathbf{b} - \mathbf{n_r})\,\bar{q}\\ \mathbf{n_w}\end{bmatrix} \tag{13}$$

$$\Leftrightarrow \dot{\mathbf{x}} = f(\mathbf{x}, \boldsymbol{\omega}_m, \mathbf{n}) \tag{14}$$

Note that the updates using the line measurements from the camera will also affect the bias estimates through the correlations between bias and quaternion.

5) Continuous-Time Error-State Model: The error state of the proposed attitude estimator includes the error in the bias and the quaternion estimate. While the bias error is defined as the vector difference between the true and the estimated bias, $\mathbf{b}$ and $\hat{\mathbf{b}}$ respectively,

$$\Delta\mathbf{b} = \mathbf{b} - \hat{\mathbf{b}} \tag{15}$$

a *multiplicative* error representation is chosen for the quaternion. Here, the attitude error is modeled as the infinitesimal rotation that causes the estimated attitude to match the true orientation. In quaternion algebra, this is expressed as

$$\bar{q} = \delta\bar{q} \otimes \hat{\bar{q}} \Leftrightarrow \delta\bar{q} = \bar{q} \otimes \hat{\bar{q}}^{-1} \tag{16}$$

Application of the small angle approximation $\delta\theta_q \simeq 0 \Rightarrow \cos(\delta\theta_q/2) \simeq 1$, $\sin(\delta\theta_q/2) \simeq \delta\theta_q/2$ leads to

$$\delta\bar{q} = \begin{bmatrix}\hat{\mathbf{k}}\sin(\delta\theta_q/2)\\ \cos(\delta\theta_q/2)\end{bmatrix} \simeq \begin{bmatrix}\hat{\mathbf{k}}\delta\theta_q/2\\ 1\end{bmatrix} = \begin{bmatrix}\frac{1}{2}\delta\boldsymbol{\theta}\\ 1\end{bmatrix} \tag{17}$$

As evident from Eq. (17), the error information is contained primarily in the tilt angle vector $\delta\boldsymbol{\theta}_{3\times1}$. Therefore the attitude uncertainty can be represented by a 3×3 covariance matrix $E[\delta\boldsymbol{\theta}\,\delta\boldsymbol{\theta}^T]$, thus circumventing the loss of rank that would arise in a 4×4 covariance matrix $E[\delta\bar{q}\,\delta\bar{q}^T]$ due to the unit quaternion constraint.

The error state vector[2] of the EKF is given by

$$\tilde{\mathbf{x}} = \begin{bmatrix}\delta\boldsymbol{\theta}\\ \Delta\mathbf{b}\end{bmatrix}_{6\times1} \tag{18}$$

[2]Notice that the state vector $\mathbf{x}$ is of dimension 7×1, whereas the error state vector $\tilde{\mathbf{x}}$ has size 6×1.

Substituting Eqs. (16), (17) in (5), and (15) in (10), we can derive the system propagation equation for the continuous-time error state [20]:

$$\begin{bmatrix}\dot{\delta\boldsymbol{\theta}}\\ \dot{\Delta\mathbf{b}}\end{bmatrix} = \begin{bmatrix}-\lfloor\hat{\boldsymbol{\omega}}\times\rfloor & -\mathbf{I}_{3\times3}\\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3}\end{bmatrix}\begin{bmatrix}\delta\boldsymbol{\theta}\\ \Delta\mathbf{b}\end{bmatrix} + \begin{bmatrix}-\mathbf{I}_{3\times3} & \mathbf{0}_{3\times3}\\ \mathbf{0}_{3\times3} & \mathbf{I}_{3\times3}\end{bmatrix}\begin{bmatrix}\mathbf{n_r}\\ \mathbf{n_w}\end{bmatrix}$$

$$\Leftrightarrow \dot{\tilde{\mathbf{x}}} = \mathbf{F}_c\cdot\tilde{\mathbf{x}} + \mathbf{G}_c\cdot\mathbf{n} \tag{19}$$

The covariance of the noise vector $\mathbf{n}$ is $E[\mathbf{n}\mathbf{n}^T] = \mathbf{Q}_c\delta(t-t')$, where $\mathbf{Q}_c$ is a block-diagonal matrix, with diagonal elements $\sigma_{r_c}^2\mathbf{I}_{3\times3}$ and $\sigma_{w_c}^2\mathbf{I}_{3\times3}$ (cf. Eqs. (11) and (12)).

6) State Propagation: For implementation on a digital computer, we need to discretize the continuous time state model. Based on Eqs. (9) and (10), the discrete-time gyroscope model can be written as

$$\boldsymbol{\omega}_k = \boldsymbol{\omega}_{m_k} - \mathbf{b}_k - \mathbf{n_r}_k \tag{20}$$

$$\mathbf{b}_{k+1} = \mathbf{b}_k + \mathbf{n_w}_k \tag{21}$$

and thus the estimated values of these quantities are computed as

$$\hat{\boldsymbol{\omega}}_{k+1|k} = \boldsymbol{\omega}_{m_{k+1}} - \hat{\mathbf{b}}_{k+1|k} \tag{22}$$

$$\hat{\mathbf{b}}_{k+1|k} = \hat{\mathbf{b}}_{k|k} \tag{23}$$

The subscript $(\cdot)_{k+1|k}$ denotes the estimate at time step $k+1$ conditioned on all available measurements up to time step k.

In order to propagate the attitude estimate, we employ the quaternion integrator of Eq. (7) with the estimated rotational velocity $\hat{\boldsymbol{\omega}}$.

7) Covariance Propagation: The error-state equation (Eq. (19)) is discretized as

$$\tilde{\mathbf{x}}_{k+1} = \boldsymbol{\Phi}_k\cdot\tilde{\mathbf{x}}_k + \mathbf{n}_d \tag{24}$$

In order to implement the discrete form of the covariance propagation equation of the EKF, we need to determine the state transition matrix $\boldsymbol{\Phi}_k$, as well as the discrete-time system noise covariance matrix $\mathbf{Q}_d$. Assuming that $\boldsymbol{\omega}$ is constant over the integration time step Δt, we can compute the state transition matrix as

$$\boldsymbol{\Phi}(t_{k+1}, t_k) = \exp\left(\int_{t_k}^{t_{k+1}}\mathbf{F}_c(\tau)\,\mathrm{d}\tau\right) \tag{25}$$

while the discrete-time system noise covariance matrix $\mathbf{Q}_d$ is computed according to

$$\mathbf{Q}_d = \int_{t_k}^{t_{k+1}}\boldsymbol{\Phi}(t_{k+1},\tau)\mathbf{G}_c(\tau)\mathbf{Q}_c\mathbf{G}_c^{\mathrm{T}}(\tau)\boldsymbol{\Phi}^{\mathrm{T}}(t_{k+1},\tau)\,\mathrm{d}\tau$$

The detailed expressions for $\boldsymbol{\Phi}$ and $\mathbf{Q}_d$ can be found in Appendix I. For clarity of notation, from now on we denote $\boldsymbol{\Phi}(t_{k+1}, t_k) = \boldsymbol{\Phi}_k$.

Following the regular EKF equations [17], we can compute the covariance of the propagated state estimate as

$$\mathbf{P}_{k+1|k} = \boldsymbol{\Phi}_k\mathbf{P}_{k|k}\boldsymbol{\Phi}_k^{\mathrm{T}} + \mathbf{Q}_d \tag{26}$$

In order to increase the accuracy of the attitude estimates, it is necessary to use *exteroceptive* measurements of features in the robot's environment, to periodically update the orientation estimates. In the application under consideration the most prominent features are the stair edges. We have therefore

developed an algorithm that processes the images recorded by an onboard camera, detects the projections of the stair edges, and employs these observations to update the attitude estimates.

8) Addressing Processing Delays: In our implementation, the gyroscope measurements are processed at a rate of 100Hz, whereas the time needed for processing each image is approximately 60msec. In order to treat the existing processing delays, we employ an approach similar to the one proposed in [24] for treating measurements that depend on previous states. In particular, at time-step k, when an image is registered, a copy of the filter state is created and added to the state vector. The error state is also duplicated, and thus the augmented error-state vector at time-step k is given by

$$\breve{\mathbf{x}}_{k|k} = \begin{bmatrix} \widetilde{\mathbf{x}}_{k|k} \\ \widetilde{\mathbf{x}}_{s_{k|k}} \end{bmatrix} \quad (27)$$

where $\widetilde{\mathbf{x}}_{s_{k|k}}$ denotes the static copy of the state, which does not evolve in time. During the time interval $[k, k+d]$, while the image is being processed, rotational velocity measurements are integrated to propagate the evolving state, while the second, static copy, remains unchanged. The benefit of this formulation is that when the measurement becomes available at time-step $k+d$, both the current state *and* the state at the time instant of the image registration are included in the augmented filter state vector. Thus, the measurement error can be expressed as a function of the augmented filter state, and the standard EKF equations can be applied for updating.

In order to correctly update the current state, the covariance matrix of the augmented filter state must also be computed. We note that, since state augmentation creates two variables that contain the exact same information (cf. Eq. (27)), these are initially fully correlated. Thus, the covariance matrix of the augmented state vector at time step k, immediately after the augmentation is performed, is:

$$\breve{\mathbf{P}}_{k|k} = \begin{bmatrix} \mathbf{P}_{k|k} & \mathbf{P}_{k|k} \\ \mathbf{P}_{k|k} & \mathbf{P}_{k|k} \end{bmatrix} \quad (28)$$

At every time step when a rotational velocity measurement is processed, the current robot state is propagated as shown in the preceding section, while the previous, static state, remains unchanged. Thus, the error propagation equation for the augmented state vector is:

$$\begin{aligned} \breve{\mathbf{x}}_{k+1|k} &= \begin{bmatrix} \mathbf{\Phi}_k & \mathbf{0}_{6\times 6} \\ \mathbf{0}_{6\times 6} & \mathbf{I}_{6\times 6} \end{bmatrix} \breve{\mathbf{x}}_{k|k} + \begin{bmatrix} \mathbf{n}_d \\ \mathbf{0}_{6\times 1} \end{bmatrix} \\ &= \breve{\mathbf{\Phi}}_k \breve{\mathbf{x}}_{k|k} + \breve{\mathbf{n}}_d \end{aligned} \quad (29)$$

and the covariance matrix of the augmented state is propagated according to

$$\begin{aligned} \breve{\mathbf{P}}_{k+1|k} &= \breve{\mathbf{\Phi}}_k \breve{\mathbf{P}}_{k|k} \breve{\mathbf{\Phi}}_k^T + \begin{bmatrix} \mathbf{Q}_d & \mathbf{0}_{6\times 6} \\ \mathbf{0}_{6\times 6} & \mathbf{0}_{6\times 6} \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{\Phi}_k \mathbf{P}_{k|k} \mathbf{\Phi}_k^T + \mathbf{Q}_d & \mathbf{\Phi}_k \mathbf{P}_{k|k} \\ \mathbf{P}_{k|k} \mathbf{\Phi}_k^T & \mathbf{P}_{k|k} \end{bmatrix} \end{aligned} \quad (30)$$

It is straightforward to show by induction that if d propagation steps take place in the time interval between image registration and the time that the line measurements become available to the filter, the covariance matrix $\breve{\mathbf{P}}_{k+d|k}$ is determined as

$$\breve{\mathbf{P}}_{k+d|k} = \begin{bmatrix} \mathbf{P}_{k+d|k} & \mathcal{F}_{k+d} \mathbf{P}_{k|k} \\ \mathbf{P}_{k|k} \mathcal{F}_{k+d}^T & \mathbf{P}_{k|k} \end{bmatrix} \quad (31)$$

where

$$\mathcal{F}_{k+d} = \prod_{i=0}^{d-1} \mathbf{\Phi}_{k+i} \quad (32)$$

$\mathbf{P}_{k+d|k}$ in Eq. (31) is the propagated covariance of the state at time-step $k+d$, which is computed by recursive application of Eq. (26).

The expression in Eq. (31) indicates that exploiting the structure of the propagation equations allows for the covariance matrix of the filter to be propagated with minimal computation. Essentially, compared to the case where only the current state is kept in the state vector, the only additional computation that needs to be performed every time a new gyroscope measurement $\boldsymbol{\omega}_{m_{k+i}}$, becomes available, is the "accumulation" of the state-transition matrices, $\mathbf{\Phi}_{k+i}$, to evaluate the term $\mathcal{F}_{k+i}$. Since only one matrix multiplication per time-step is necessary, this can be performed very efficiently.

9) State and Covariance Update: In this section, we describe the measurement model we employ for performing EKF updates. In each of the images recorded by the camera, a straight-line detection algorithm is applied (cf. Appendix II) to obtain measurements of the projections of the stair edges in the image. In the following we assume, without loss of generality, that all quantities are expressed with respect to a normalized camera frame with unit focal length.

The output of the line detection algorithm is a set of M line measurements, given by

$$\boldsymbol{\ell}_{m_j} = \boldsymbol{\ell}_j + \mathbf{n}_j, \quad j = 1 \dots M \quad (33)$$

where $\boldsymbol{\ell}_{m_j}$ is the measured line, $\boldsymbol{\ell}_j$ is the true line on the image plane, and $\mathbf{n}_j$ is a 3×1 noise vector, with covariance matrix $\mathbf{R}_j$ (cf. Eq. (80)). A line is defined by its polar representation, i.e.,

$$\boldsymbol{\ell}_j = \begin{bmatrix} \cos\phi_j & \sin\phi_j & -\rho_j \end{bmatrix}^T \quad (34)$$

where (ϕ_j, ρ_j) are the line parameters, representing the orientation and magnitude of the line's normal vector ($\overrightarrow{OP_j}$ in Fig. 5). A point $\mathbf{p}$ with homogeneous image coordinates $p = [u \quad v \quad 1]^T$ lies on the line $\boldsymbol{\ell}_j$ if it satisfies the equation

$$u\cos\phi_j + v\sin\phi_j - \rho_j = 0 \Rightarrow \mathbf{p}^T \boldsymbol{\ell}_j = 0 \quad (35)$$

We now derive a geometric constraint relating the measurements of the lines on the image plane with the robot's attitude. Let O denote the principal point of the image plane, F denote the focal point of the camera, and $\mathbf{u}_j = [\sin\phi_j \quad -\cos\phi_j \quad 0]^T$ be a (free) unit vector along the line on the image plane. From Fig. 5 we observe that the vectors $\mathbf{u}_j$ and $\overrightarrow{FP_j} = \overrightarrow{FO} + \overrightarrow{OP_j} = [\rho_j\cos\phi_j \quad \rho_j\sin\phi_j \quad 1]^T$ define a plane that contains the observed line (i.e., it contains the vector $\mathbf{e}_i$). The normal vector to this plane is defined as

$$\overrightarrow{FP_j} \times \mathbf{u}_j = \begin{bmatrix} \rho_j\cos\phi_j \\ \rho_j\sin\phi_j \\ 1 \end{bmatrix} \times \begin{bmatrix} \sin\phi_j \\ -\cos\phi_j \\ 0 \end{bmatrix} = \begin{bmatrix} \cos\phi_j \\ \sin\phi_j \\ -\rho_j \end{bmatrix} = \boldsymbol{\ell}_j \quad (36)$$

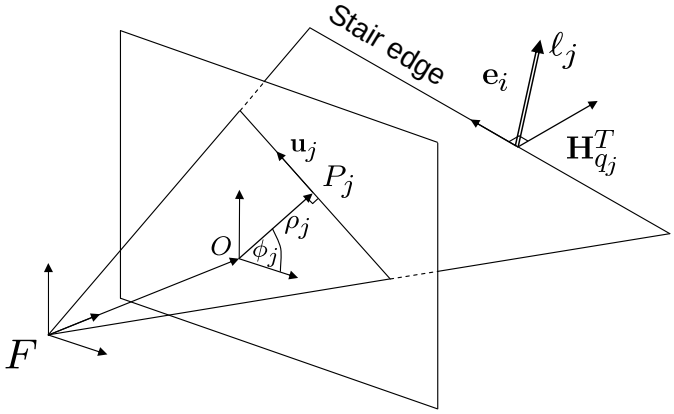


Fig. 5. Spatial relationship between the observed unit vector $\mathbf{e}_i$, the line on the image plane with unit vector $\mathbf{u}_j$, the line measurement vector $\boldsymbol{\ell}_j$, and the observation jacobian $\mathbf{H}_{q_j}^T$. The focal point of the camera is denoted as F, and the point on the line with minimum distance ρ_j to the principal point O as P_j.

Since the vector $\mathbf{e}_i$ is contained in the plane with normal vector $\boldsymbol{\ell}_j$, we obtain $\mathbf{e}_i \perp \boldsymbol{\ell}_j$, and thus

$$\left({}^C\mathbf{e}_i\right)^T \boldsymbol{\ell}_j = 0 \Rightarrow \mathbf{e}_i^T \mathbf{C}^T({}^C_G\bar{q}_s)\boldsymbol{\ell}_j = 0 \tag{37}$$

where $\mathbf{C}({}^C_G\bar{q}_s)$ is the rotational matrix that transforms vectors from the global frame to the camera frame at the time instant that the measurement was recorded.[3]

The expression in Eq. (37) defines the geometric constraint that relates the vector $\boldsymbol{\ell}_j$, associated with a line projection in the image, to the global unit vector $\mathbf{e}_i$. This expression is exact for the *true* quaternion representing the rotation between the global and the camera frame, and for the *true* projection of a line on the image. These quantities, however, are not available in practice. Instead, a noise-corrupted measurement of the line equation (cf. Eq. (33)) and the estimate of the robot's orientation at the time of the image registration, $\hat{\bar{q}}_s$, are known. Due to errors in the line measurement and the robot's orientation estimate, when the constraint of Eq. (37) is evaluated using the estimates of the corresponding quantities, a *residual* arises:

$$\begin{aligned} r_j &= z_j - \widehat{z}_j \\ &= \mathbf{e}_i^T \mathbf{C}^T({}^C_G\hat{\bar{q}}_s)\boldsymbol{\ell}_{m_j} - 0 \\ &= \mathbf{e}_i^T \mathbf{C}^T({}^C_G\hat{\bar{q}}_s)\boldsymbol{\ell}_{m_j} \end{aligned} \tag{38}$$

where $\mathbf{C}({}^C_G\hat{\bar{q}}_s)$ is the estimated rotation matrix between the camera and the global frame.

In order to express the residual as a function of the errors in the robot attitude estimate, $\hat{\bar{q}}_s$, and in the line measurement, $\boldsymbol{\ell}_{m_j}$, we denote the quaternion representing the rotation between the camera frame and the robot frame as ${}^C_L\bar{q}$, thus obtaining:

$$\mathbf{C}({}^C_G\hat{\bar{q}}_s) = \mathbf{C}({}^C_L\bar{q})\mathbf{C}(\hat{\bar{q}}_s) \tag{39}$$

[3]An alternative way to derive the constraint in Eq. (37) is to note that the vanishing point along the vector $\mathbf{e}_i$ projects on the point $\mathbf{p} = \mathbf{C}({}^C_G\bar{q}_s)\mathbf{e}_i$ in the image plane. Since this point lies on the line $\boldsymbol{\ell}_j$ in the image, Eq. (37) follows directly from application of Eq. (35).

and

$$\begin{aligned} \mathbf{C}({}^C_G\bar{q}_s) &= \mathbf{C}({}^C_L\bar{q})\mathbf{C}(\bar{q}_s) \\ &= \mathbf{C}({}^C_L\bar{q})\mathbf{C}(\delta\bar{q}_s)\mathbf{C}(\hat{\bar{q}}_s) \end{aligned} \tag{40}$$

where $\delta\bar{q}_s = \bar{q}_s \otimes \hat{\bar{q}}_s^{-1}$ represents the error quaternion at the time instant of the image registration. Thus we can rewrite Eq. (38) as:

$$\begin{aligned} r_j &= \mathbf{e}_i^T \mathbf{C}^T({}^C_G\hat{\bar{q}}_s)\boldsymbol{\ell}_{m_j} - \mathbf{e}_i^T \mathbf{C}^T({}^C_G\bar{q}_s)\boldsymbol{\ell}_j \\ &= \mathbf{e}_i^T \mathbf{C}^T(\hat{\bar{q}}_s)\left(\mathbf{C}^T({}^C_L\bar{q})\boldsymbol{\ell}_{m_j} - \mathbf{C}^T(\delta\bar{q}_s)\mathbf{C}^T({}^C_L\bar{q})\left(\boldsymbol{\ell}_{m_j} - \mathbf{n}_j\right)\right) \end{aligned}$$

By employing the small angle approximation [19]:

$$\mathbf{C}(\delta\bar{q}_s) \simeq \mathbf{I}_{3\times 3} - \lfloor \delta\boldsymbol{\theta}_\mathbf{s} \times \rfloor \quad , \tag{41}$$

and ignoring quadratic error terms, we obtain the following expression for the residual:

$$\begin{aligned} r_j &\simeq \mathbf{e}_i^T \mathbf{C}^T(\hat{\bar{q}}_s)\lfloor \mathbf{C}^T({}^C_L\bar{q})\boldsymbol{\ell}_{m_j} \times \rfloor \delta\boldsymbol{\theta}_s + \mathbf{e}_i^T \mathbf{C}^T(\hat{\bar{q}}_s)\mathbf{C}^T({}^C_L\bar{q})\mathbf{n}_j \\ &= \begin{bmatrix}\mathbf{0}_{1\times 6} & \mathbf{H}_{s_j}\end{bmatrix} \begin{bmatrix} \delta\boldsymbol{\theta}_{k+d|k} \\ \Delta\mathbf{b}_{k+d|k} \\ \delta\boldsymbol{\theta}_s \\ \Delta\mathbf{b}_s \end{bmatrix} + \boldsymbol{\Gamma}_j \mathbf{n}_j \\ &= \mathbf{H}_j \check{\mathbf{x}} + \boldsymbol{\Gamma}_j \mathbf{n}_j \end{aligned} \tag{42}$$

where we have denoted

$$\begin{aligned} \mathbf{H}_{s_j} &= \begin{bmatrix}\mathbf{e}_i^T \mathbf{C}^T(\hat{\bar{q}}_s)\lfloor \mathbf{C}^T({}^C_L\bar{q})\boldsymbol{\ell}_{m_j} \times \rfloor & \mathbf{0}_{1\times 3}\end{bmatrix} \\ &= \begin{bmatrix}\mathbf{H}_{q_j} & \mathbf{0}_{1\times 3}\end{bmatrix} \end{aligned} \tag{43}$$

$$\boldsymbol{\Gamma}_j = \mathbf{e}_i^T \mathbf{C}^T(\hat{\bar{q}}_s)\mathbf{C}^T({}^C_L\bar{q}) \tag{44}$$

Eq. (42) defines the linearized residual error equation for one line, that results from the projection of a known unit vector $\mathbf{e}_i$. If multiple lines are detected in an image, the residuals corresponding to all lines can be stacked to form a residual vector, which can consequently be used for performing EKF updates.

In our implementation, we are employing measurements of the projections of the stair edges, which are parallel to the global y-axis. Although straight lines other than stair edges can generally also be detected in the images (cf. Fig. 12), it is not easy to determine the corresponding global unit vector. In order to discard any measurements that do not belong to lines parallel to the global y-axis (unit vector $\mathbf{e}_2$), we perform a gating test with every detected line, prior to using it for state updates. In particular, for each line we compute the residual r_j using Eq. (38), and require that it satisfies the Mahalanobis distance test:

$$\frac{r_j^2}{\mathbf{H}_{s_j}\mathbf{P}_{k|k}\mathbf{H}_{s_j}^T + \boldsymbol{\Gamma}_j \mathbf{R}_j \boldsymbol{\Gamma}_j^T} < \gamma \tag{45}$$

where γ is equal to the 99-percentile of the χ_1^2 distribution (i.e., $\gamma = 6.63$). Fig. 6 shows an example image recorded by the robot's camera. The lines that pass (fail) the Mahalanobis distance test are superimposed with solid (dashed) lines. Note that the accepted lines do not necessarily belong to the stair steps, but they are all parallel to the global y-axis.

In order to perform the EKF updates, all lines that pass the gating test are used to define the $M \times 1$ residual vector

$$\begin{aligned} \mathbf{r} &= \mathbf{H}\check{\mathbf{x}} + \mathbf{\Gamma n} \\ &= \begin{bmatrix} \mathbf{0}_{M\times 6} & \mathbf{H}_s \end{bmatrix} \check{\mathbf{x}} + \mathbf{\Gamma n} \end{aligned} \quad (46)$$

where $\mathbf{r}$ is the vector with elements r_j (Eq. (38)), $\mathbf{H}_s$ is a matrix with block rows $\mathbf{H}_{s_j}$ (Eq. (43)), $\mathbf{\Gamma}$ is a block-diagonal matrix with elements $\mathbf{\Gamma}_j$ (Eq. (44)), and $\mathbf{n}$ is an error vector with block elements $\mathbf{n}_j$ (Eq. (33)). Since the errors in the measurements of the individual lines are independent, the covariance matrix of $\mathbf{n}$ is a block diagonal matrix, $\mathbf{R}$, with diagonal elements $\mathbf{R}_j$ (computed using Eq. (80) in Appendix II).

Eq. (46) defines the innovation of the EKF update. The covariance matrix of the innovation is given by

$$\begin{aligned} \mathbf{S} &= \mathbf{H}\check{\mathbf{P}}_{k+d|k}\mathbf{H}^T + \mathbf{\Gamma R \Gamma}^T \\ &= \mathbf{H}_s\mathbf{P}_{k|k}\mathbf{H}_s^T + \mathbf{\Gamma R \Gamma}^T \end{aligned} \quad (47)$$

and thus the Kalman gain matrix is determined as:

$$\mathbf{K} = \check{\mathbf{P}}_{k+d|k}\mathbf{H}^T\mathbf{S}^{-1} = \begin{bmatrix} \mathbf{K}_{k+d} \\ \mathbf{K}_s \end{bmatrix} \quad (48)$$

where $\mathbf{K}_{k+d}$ is the Kalman gain for the current state, and $\mathbf{K}_s$ is the gain for the static state (i.e., for the state at the time instant of image registration). It is important to note that the static copy of the state does *not* have to be updated, as only the current attitude is necessary for motion control. Therefore, evaluation of $\mathbf{K}_s$ is not necessary, and is omitted to reduce computations. The block element of $\mathbf{K}$ corresponding to the current state is given by (cf. Eqs. (31), (47) and (48)):

$$\mathbf{K}_{k+d} = \mathcal{F}_{k+d}\mathbf{P}_{k|k}\mathbf{H}_s^T(\mathbf{H}_s\mathbf{P}_{k|k}\mathbf{H}_s^T + \mathbf{\Gamma R \Gamma}^T)^{-1}$$

The current error-state correction is computed as

$$\begin{bmatrix} \widehat{\delta\boldsymbol{\theta}}_{k+d} \\ \widehat{\Delta\mathbf{b}}_{k+d} \end{bmatrix} = \mathbf{K}_{k+d}\mathbf{r}$$

The update for the quaternion is given by

$$\hat{\bar{q}}_{k+d|k+d} = \widehat{\delta\bar{q}}_{k+d} \otimes \hat{\bar{q}}_{k+d|k} \quad (49)$$

where

$$\widehat{\delta\bar{q}}_{k+d} = \begin{bmatrix} \frac{1}{2}\widehat{\delta\boldsymbol{\theta}}_{k+d} \\ \sqrt{1 - \frac{1}{4}\widehat{\delta\boldsymbol{\theta}}_{k+d}^T\widehat{\delta\boldsymbol{\theta}}_{k+d}} \end{bmatrix}$$

The update for the bias estimate is simply

$$\hat{\mathbf{b}}_{k+d|k+d} = \hat{\mathbf{b}}_{k+d|k} + \widehat{\Delta\mathbf{b}}_{k+d} \quad (50)$$

Finally, the covariance matrix for the current state is updated as:

$$\mathbf{P}_{k+d|k+d} = \mathbf{P}_{k+d|k} - \mathbf{K}_{k+d}\mathbf{S}\mathbf{K}_{k+d}^T \quad (51)$$

For clarity, we present the steps of the attitude estimation algorithm in Table 1.

Algorithm 1 Attitude Estimation Kalman filter

Propagation: Every time a rotational velocity measurement is received:

- propagate the current state estimate using the estimated rotational velocities at the last two time steps in Eq. (7), and Eqs. (22) and (23)
- propagate the covariance of the current filter state, using Eq. (26)
- if a copy of a previous state is present (i.e., if an image is currently being processed), compute the matrix $\mathcal{F}_{k+i}$ using Eq. (32)

Copying the state: Every time an image is recorded:

- create a copy of the current quaternion estimate and the current state covariance matrix

Update: Every time line measurements from an image become available:

- perform gating tests for all detected lines (Eq. (45))
- use the lines that pass the gating test to update the state using Eqs. (49) and (50)
- update the covariance of the current state vector using Eq. (51)
- discard the static copy of the state

B. Estimating the distance ratio d_L/d_R

In order to avoid collisions with the boundaries of the stairs, the centering controller requires an estimate of the robot's distance to the walls. Given the projections of the stair edges, it is possible to estimate the ratio of the distances from the *camera* to the left and right ends of the stairs. We note that this ratio will in general differ from the ratio of the distances of the *robot's* center from the ends of the stairs. However, for small steering angles this difference is not significant, and we found that it does not hinder the controller's performance.

The 3D coordinates of two points that lie on the left and right ends of a stair edge are given respectively by:

$${}^G\mathbf{p}_L = \begin{bmatrix} x_o \\ w \\ z_o \end{bmatrix} \quad \text{and} \quad {}^G\mathbf{p}_R = \begin{bmatrix} x_o \\ 0 \\ z_o \end{bmatrix} \quad (52)$$

where the width of the stairs is denoted by w, and the coordinates x_o and z_o can be arbitrary (cf. Fig. 4). The projective image coordinates of the projection of the left endpoint on the image are determined by:

$$\begin{aligned} \mathbf{p}_{Lp} &= \frac{1}{c_L}\begin{bmatrix} \mathbf{C}({}^C_G\bar{q}_s) & {}^C\mathbf{p}_G \end{bmatrix}\begin{bmatrix} {}^G\mathbf{p}_L \\ 1 \end{bmatrix} \\ &= \frac{1}{c_L}\begin{bmatrix} \mathbf{C}({}^C_G\bar{q}_s) & -\mathbf{C}({}^C_G\bar{q}_s){}^G\mathbf{p}_C \end{bmatrix}\begin{bmatrix} {}^G\mathbf{p}_L \\ 1 \end{bmatrix} \\ &= \frac{1}{c_L}\mathbf{C}({}^C_G\bar{q}_s)\left({}^G\mathbf{p}_L - {}^G\mathbf{p}_C\right) \end{aligned} \quad (53)$$

where the vector ${}^G\mathbf{p}_C = [x_c \;\; y_c \;\; z_c]^T$ denotes the position of the camera in the global coordinate frame, and c_L is an

arbitrary nonzero scalar. From the last expression we obtain:

$$\begin{bmatrix} x_o - x_c \\ w - y_c \\ z_o - z_c \end{bmatrix} = c_L \mathbf{C}^T({}^C_G\bar{q}_s)\mathbf{p}_{Lp} \quad (54)$$

By employing similar derivations for the right endpoint of the stair edge, we obtain

$$\begin{bmatrix} x_o - x_c \\ -y_c \\ z_o - z_c \end{bmatrix} = c_R \mathbf{C}^T({}^C_G\bar{q}_s)^T\mathbf{p}_{Rp} \quad (55)$$

for some nonzero multiplicative constant c_R.

At this point, we note that the distance of the camera from the left side of the stairs is $d_L = w - y_c$, while the distance from the right side is $d_R = y_c$. Moreover, an estimate for the right-hand side of Eqs. (54) and (55), up to a multiplicative constant, can be computed by employing the estimate for the camera attitude, $\mathbf{C}({}^C_G\hat{\bar{q}}_s) = \mathbf{C}({}^C_L\bar{q}_s)\mathbf{C}(\hat{\bar{q}}_s)$ and the measured image coordinates of the endpoints of the line, $\mathbf{p}_{Rp_m}$ and $\mathbf{p}_{Lp_m}$. Thus, if the multiplicative constants in Eqs. (54) and (55) were known, it would be possible to directly estimate the quantities $d_R = y_c$ and $d_L = w - y_c$ from these equations. However, due to the scale uncertainty introduced by the use of a single camera, only the ratio of the multiplicative constants can be computed. By noting that the first and third elements of the vectors in the left-hand side of Eqs. (54) and (55) are equal, we can estimate the ratio c_L/c_R as

$$\frac{c_L}{c_R} = \sqrt{\frac{\left(\mathbf{e}_1^T\mathbf{C}^T({}^C_G\hat{\bar{q}}_s)\mathbf{p}_{Rp_m}\right)^2 + \left(\mathbf{e}_3^T\mathbf{C}^T({}^C_G\hat{\bar{q}}_s)\mathbf{p}_{Rp_m}\right)^2}{\left(\mathbf{e}_1^T\mathbf{C}^T({}^C_G\hat{\bar{q}}_s)\mathbf{p}_{Lp_m}\right)^2 + \left(\mathbf{e}_3^T\mathbf{C}^T({}^C_G\hat{\bar{q}}_s)\mathbf{p}_{Lp_m}\right)^2}}$$

and thus an estimate for the ratio of the distances d_L/d_R can be computed as

$$\frac{d_L}{d_R} = \frac{w - y_c}{y_c} = \frac{c_L}{c_R}\left|\frac{\mathbf{e}_2^T\mathbf{C}^T({}^C_G\bar{q}_s)\mathbf{p}_{Lp_m}}{\mathbf{e}_2^T\mathbf{C}^T({}^C_G\bar{q}_s)\mathbf{p}_{Rp_m}}\right| \quad (56)$$

In every processed image, an estimate for the ratio d_L/d_R is computed from each of the lines that are found to be parallel to the global y-axis. Our experiments have shown that these estimates can vary significantly within an image, due to the fact that the localization of the lines' endpoints is not very reliable. Several factors contribute to this: (i) Due to the properties of light reflection, the corners between the stairs and the adjacent walls are illuminated less than the rest of the stairs. (ii) Due to the accumulation of dirt and the effects of use, the ends of stair edges often have different appearance than the center. (iii) The robot undergoes rapid rotations about its x-axis (which coincides with the camera z-axis), as a result of the tracks' interaction with the steps (cf. Fig. 9). These rotations result in image blurring, that is more significant at larger angles from the optical axis. (iv) The camera lens exhibits vignetting, thus resulting in lower contrast towards the periphery of the images.

The above discussion indicates that it is necessary to employ a robust scheme for fusing the ratio estimates of different lines, to ensure that spurious measurements do not cause large fluctuations in the robot's ratio estimate. In order to discard conspicuous outliers, we do not consider lines that are shorter than lines above them in an image. This constraint arises from the geometry of the projection model, which dictates that lines that are closer to the robot (and thus lower in the image) should appear larger[4]. Moreover, we employ a median filter to compute the median ratio estimate from all the lines detected in the last five image frames. Since the median is not sensitive to the existence of a small percentage of outliers in the data, the ratio estimates we obtain are more robust. We note at this point that the delay introduced by this temporal averaging is not significant: since images are processed at a rate of 15Hz, any large change in the true ratio of distances (for example, due to large slippage), will be detected on average in less than 0.2sec.

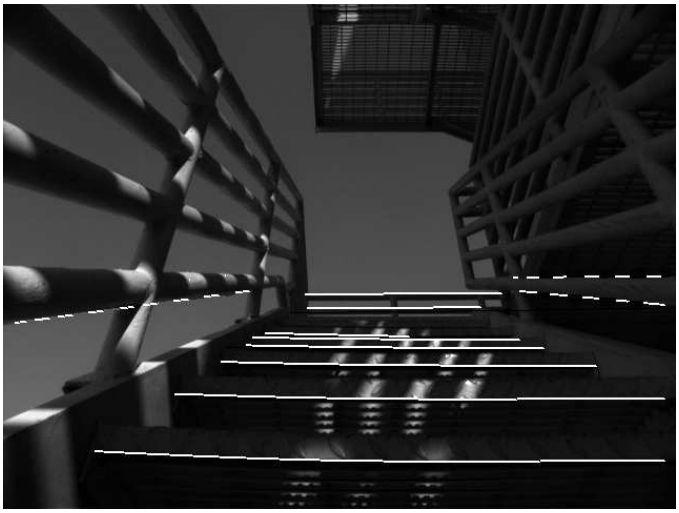


Fig. 6. An example image recorded by the camera, with the detected lines that passed the Mahalanobis test superimposed as solid lines. The dashed lines are those discarded by the gating test.

IV. Motion Control

A. Overview

The two main objectives of the stair-climbing control algorithm are: (i) maximize the time that the robot is heading directly up the stairs, and (ii) keep the vehicle away from the staircase boundaries. The first goal is necessitated primarily by the observation that the actual stair-climbing speed is significantly affected by the robot heading. Specifically, even when both track motors are commanded to rotate at the same rate, the actual linear and rotational velocities of the vehicle depend on the angle between the track cleats and the stair steps. When the cleats are parallel to the stair edges, both tracks exert maximum and approximately equal forces on the steps which results in efficient stair climbing at high speed. In contrast, if the cleats engage the stair edges at a large angle, the track-surface interaction becomes highly nonlinear and difficult to model. This is primarily due to the elasticity of the tracks, the time-varying friction coefficients, and the rapid and unpredictable changes in the percentage of the tracks' surface

[4]An exception applies for lines that extend up to the end of the image. In our implementation, these lines are not discarded by this rule.

that is in contact with the stairs. This complex interaction causes disturbances in the motion of the vehicle (intense track slip, large lateral velocities and rotational accelerations) whose magnitude increases with the robot velocity. This situation can lead to uncontrollable motion and failure due to collisions or even toppling of the vehicle.[5]

Under ideal conditions of operation, a *heading* controller designed so as to minimize the heading error estimated by the EKF (Section III) should be sufficient for guaranteeing that the robot will travel straight up the stairs. As long as the vehicle starts at the center of the stairs, it should be expected that it will finish close to the staircase centerline. However, the trajectory disturbances due to the highly dynamic motion profile, often cause the robot to move towards the boundaries of the staircase. In order to avoid collisions with the walls or the stair railing, it is necessary to be able to detect when the vehicle approaches the stair sides and provide appropriate correction. To this end, we have designed a *centering* controller, which, given the ratio of the distances to the stair boundaries (Section III-B), changes the reference signal (heading direction) of the heading controller and brings the vehicle closer to the centerline. This two-tiered approach to the design of the stair-climbing controller system (cf. Fig. 2) is described in detail in the following two sections.

At this point, we should note that the centering controller computes a heading direction θ_r, every time it receives an estimate of the distance ratio, d_L/d_R. These estimates become available asynchronously from the image processing algorithm at a rate $f_c \simeq$15Hz. The heading controller receives as input (i) the heading reference direction θ_r dictated by the centering controller, (ii) the yaw, $\hat{\theta}$, and pitch, $\hat{\alpha}$, estimates from the EKF, and (iii) the desired linear velocity of the vehicle, V, specified by the user. The output of the heading controller is the commanded rotational velocity, ω_d, of the robot. Although estimates of the vehicle's heading are provided from the EKF at a rate of f_e=100Hz, the heading controller operates at f_h =30Hz. This rate has been determined experimentally to be fast enough to react to the dynamics of the vehicle while placing reasonable computational demands on the system. The input and output signals for both controllers are depicted in Fig. 2.

B. Centering Controller

As previously mentioned, the optimal heading direction for a stair-climbing vehicle is $\theta_r = 0$. However, when the robot approaches the staircase boundaries, the threat of collision requires the centering controller to deviate from the optimal heading direction and steer the robot away from the stair sides. The information available to the centering controller for predicting whether the robot is outside a "safe zone" around the centerline, is the ratio of the distances d_L/d_R to the left and right boundaries of the staircase (Fig. 7). Since the ratio

[5]These observations are corroborated by numerous trials of human operators attempting to remotely control the vehicle up the stairs. The most common modes of failure are: (i) collision with the staircase boundaries, (ii) toppling of the vehicle. The main reasons for these events were high lateral velocities and/or sudden changes of the motion direction that caused the vehicle to align parallel to the stair edges.

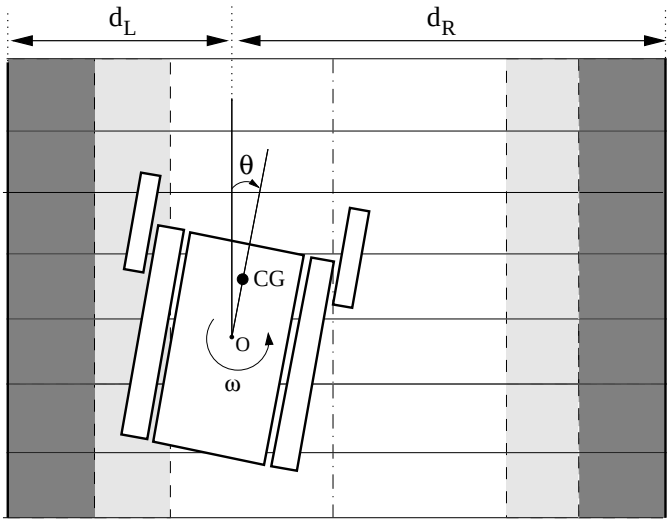


Fig. 7. Diagram of vehicle on the stairs. CG is the center of gravity, O is the center of rotation, θ is the heading direction, and d_L, d_R are the distances to the left and right of the stairs, respectively. Note in this plot that the dark-grey regions correspond to the "non-safe" areas close to staircase boundaries, while the white and light-gray regions are considered as "safe" areas. When the vehicle steers away from the stair ends, at a commanded angle $\theta_r = \theta_d$, it needs to pass through the light-grey area and move within the white region before the heading controller switches its reference signal θ_r back to the nominal heading direction of 0 degrees.

is a non-symmetric function of the robot location relative to the staircase centerline, the centering controller uses instead as input the normalized ratio $\delta = \min(d_L/d_R, d_R/d_L)$, $0 \leq \delta \leq 1$. Additionally, the sign value $s_\delta = \text{sign}(d_L/d_R - 1)$ is computed to determine the direction of the deviation from zero heading. The output of the centering controller is the reference signal θ_r provided to the heading controller (Fig. 2). The centering controller is implemented as a step function with hysteresis:

$$\theta_r = \begin{cases} 0, & \delta \geq \delta_c \\ s_\delta \cdot \theta_d, & \delta < \delta_c \end{cases} \quad (57)$$

where $\theta_d = 10^{\text{o}}$ is the magnitude of the direction change, and $\delta_c = \frac{3}{7}$ ($\delta_c = \frac{4}{7}$) is the normalized distance ratio threshold for detecting when the robot leaves (enters) the safe region around the stair centerline. Note that the threshold δ_c receives different values (hysteresis when switching between regions) depending on the direction the normalized ratio δ approaches these from. This is necessary so as to avoid oscillations of the reference signal θ_r on the region boundary. The values of θ_d and δ_c have been determined experimentally in order to minimize the disturbances on the vehicle motion and the probability of collision with the stair boundaries.

C. Heading Controller

In order to ensure that the vehicle will follow the heading direction dictated by the centering controller, a model-based heading controller has been designed. In what follows, we describe the system model employed for this purpose and the derived state-feedback controller.

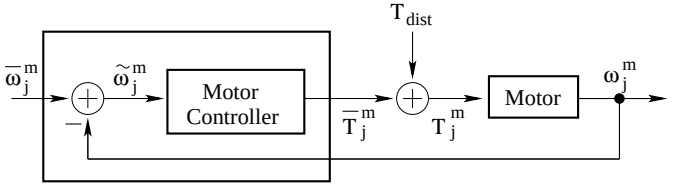


Fig. 8. Motor controller and motor block diagram.

1) System Model: A dynamics-based model of the vehicle is developed in order to design a heading controller for use during stair climbing. A detailed description of modeling techniques for tracked vehicles is presented in [25] and [26]. In this work, we have approximated the dynamics of the robot climbing stairs as a second-order linear system. This approximation does not invalidate the model; it limits though the range of application of the designed heading controller to small angles ($|\theta| < 30^{\circ}$) of robot heading direction. The main advantage of this linearized model is that it allows for the use of formal control-system design techniques when designing the heading controller [27].

As shown in Fig. 7, the center of gravity (CG) of the vehicle used in our implementation is above its center of rotation O. The equation that describes the rotation of the vehicle in a plane defined by the stair edges is

$$I_z\dot{\omega} = T_O + mgd_{CG}\sin\alpha\sin\theta - M_r \tag{58}$$

where $\dot{\omega} = \ddot{\theta}$ is the rotational acceleration, θ is the heading direction, and T_O is the torque exerted by the motors about the vehicle's center of rotation, O. The parameters in the above equation are: (i) I_z is the moment of inertia about the z-axis, computed by weighing the individual subcomponents of the vehicle and measuring their location relative to O, (ii) m is the mass of the robot, (iii) g is the magnitude of the gravitational acceleration, (iv) d_{CG} is the distance of the CG from O, (v) α is the inclination of the stairs, and (vi) M_r is the rotational resistance. This last parameter is computed as $M_r = \mu mg\cos\alpha L/8$, where L is the length of the tracks, and μ is the coefficient of lateral resistance, estimated from experimental data as in [28]. For small values of the heading direction ($\sin\theta \simeq \theta$), Eq. (58) can be approximated by the following equation:

$$I_z\ddot{\theta} \;=\; T_O + mgd_{CG}\sin\alpha\;\theta - M_r \tag{59}$$

In this last expression, the torque, T_O, on the robot body is computed as:

$$T_O = (F_R - F_L)b/2 \tag{60}$$

where b is the distance between the tracks and F_R (F_L) is the force exerted by the right (left) track of the vehicle on the steps. These forces are related to the corresponding *motor* torques T_R^m and T_L^m by the following expressions:

$$F_R = \frac{n_g}{r_s}T_R^m \quad , \quad F_L = \frac{n_g}{r_s}T_L^m \tag{61}$$

where r_s is the radius of the sprocket that drives each track and n_g is the gear ratio between the motor and the sprocket. Substituting from Eq. (61) in Eq. (60), it is:

$$T_O = (T_R^m - T_L^m)\,\frac{n_g b}{2r_s} \tag{62}$$

The commanded motor torque $T_j^m, j \in \{R, L\}$ is the output of the motor controller (cf. Fig. 8) which is modelled as a PD controller with characteristic function $h_{mc}(s) = k_p + k_d s$, and input the difference $\tilde{\omega}_j^m$ between the desired $\bar{\omega}_j^m$ and the actual ω_j^m rotational velocity of the motor. Since the response of the motor controller is extremely fast compared to the vehicle dynamics, the relationship between T_j^m and $\tilde{\omega}_j^m$ can be approximated as:

$$T_j^m = k_{mc}\;\tilde{\omega}_j^m = k_{mc}\left(\bar{\omega}_j^m - \omega_j^m\right)\;,\;\; j \in \{R, L\} \tag{63}$$

Applying the final value theorem to $h_{mc}(s)$, it can be shown that $k_{mc} = k_p$, which is known from the motor specifications. Substituting from Eq. (63) to Eq. (62), it is:

$$T_O = \frac{k_{mc}n_g b}{2r_s}\left(\tilde{\omega}_R^m - \tilde{\omega}_L^m\right) \tag{64}$$

The motor rotational velocity ω_j^m is given by:

$$\omega_j^m = V_j\frac{n_g}{r_s}\;,\;\; j \in \{R, L\}$$

where V_j is the linear velocity of the corresponding track, and n_g and r_s are defined as before. Employing this last expression and the kinematic relationship between the linear velocities of the two tracks and the rotational velocity of the robot, i.e.,

$$\omega = \left(V_R - V_L\right)/b\;,$$

it is readily shown that $\omega_R^m - \omega_L^m = \frac{n_g}{r_s}\omega$, $\bar{\omega}_R^m - \bar{\omega}_L^m = \frac{n_g}{r_s}\bar{\omega}$, and thus

$$\tilde{\omega}_R^m - \tilde{\omega}_L^m = \frac{n_g}{r_s}\tilde{\omega} \tag{65}$$

where $\tilde{\omega} = \bar{\omega} - \omega$ is defined as the difference between the commanded, $\bar{\omega} = \dot{\bar{\theta}}$, and the actual, $\omega = \dot{\theta}$, rotational velocity of the vehicle body. Substituting from Eqs. (64) and (65) in Eq. (59), we have:

$$I_z\ddot{\theta} \;=\; \frac{k_{mc}}{2}\left(\frac{n_g b}{r_s}\right)^2(\dot{\bar{\theta}} - \dot{\theta}) + mgd_{CG}\sin\alpha\;\theta - M_r$$

Rearranging the terms in this last equation and making the following substitutions

$$k_v = \frac{k_{mc}}{2I_z}\left(\frac{n_g b}{r_s}\right)^2,\;\; k_g = \frac{mgd_{CG}\sin\alpha}{I_z},\;\; \omega_d = \dot{\bar{\theta}} - \frac{M_r}{k_v I_z}$$

we have:

$$\ddot{\theta} = -k_v\;\dot{\theta} + k_g\;\theta + k_v\;\omega_d$$

This model can be written in standard state-space form as:

$$\begin{aligned}\begin{bmatrix}\dot{\theta}\\ \ddot{\theta}\end{bmatrix} \;&=\; \begin{bmatrix}0 & 1\\ k_g & -k_v\end{bmatrix}\begin{bmatrix}\theta\\ \dot{\theta}\end{bmatrix} + \begin{bmatrix}0\\ k_v\end{bmatrix}\omega_d \Rightarrow\\ \dot{\mathbf{x}}(t) \;&=\; \mathbf{A}\;\mathbf{x}(t) + \mathbf{b}\;u(t)\end{aligned} \tag{66}$$

2) Controller Design: Once the state-space model (cf. Eq. (66)) is developed, a number of techniques can be employed to design the controller. In this work, we have selected a pole placement approach which has the advantage of being able to explicitly specify the resulting dynamics of the controlled system within the constraints of the actuators [27]. The result of this design is a control law expressed as:

$$u(t) = -\mathbf{k}^T \mathbf{x}(t)$$

where $\mathbf{k}$ is the vector of the controller gains.

A few modifications to Eq. (66) are required before applying the pole placement design method. The first of these is to discretize it at a rate equal to that of the controller. As mentioned before, a heading control rate of f_h=30Hz was determined sufficient for reacting to the vehicle dynamics. The discrete-time form of Eq. (66) is:

$$\mathbf{x}(k+1) = \mathbf{A}_d \ \mathbf{x}(k) + \mathbf{b}_d \ u(k) \tag{67}$$

where $\mathbf{A}_d$ and $\mathbf{b}_d$ are the equivalent discrete-time state and and input matrices.

The second modification is the augmentation of the state vector with a heading-error integral term $\tilde{x}_I$, and the addition of a reference signal $\theta_r(k)$, i.e.,

$$\begin{aligned} \begin{bmatrix} \tilde{x}_I(k+1) \\ \mathbf{x}(k+1) \end{bmatrix} &= \begin{bmatrix} 1 & \mathbf{h}^T \\ \mathbf{0} & \mathbf{A}_d \end{bmatrix} \begin{bmatrix} \tilde{x}_I(k) \\ \mathbf{x}(k) \end{bmatrix} + \begin{bmatrix} 1 \\ \mathbf{0} \end{bmatrix} \theta_r(k) \\ &\quad + \begin{bmatrix} 0 \\ \mathbf{b}_d \end{bmatrix} u(k) \\ \Rightarrow \breve{\mathbf{x}}(k+1) &= \breve{\mathbf{A}}_d \ \breve{\mathbf{x}}(k) + \breve{\mathbf{c}}_d \ \theta_r(k) + \breve{\mathbf{b}}_d \ u(k) \end{aligned} \tag{68}$$

with $\mathbf{h}^T = [-1 \ 0]$. This extra state term $\tilde{x}_I$ is required so as to eliminate any steady-state error that may occur in the system due to disturbances caused by the unmodeled dynamics of the interaction between the vehicle tracks and the stair steps. The reference signal $\theta_r(k)$ is included in this last equation in order to allow for the centering controller to modify the system behavior by changing the heading direction of the vehicle when the robot moves close to, or away from, the staircase boundaries.

The design of the heading feedback control law, $u(k) = -\breve{\mathbf{k}}_d \ \breve{\mathbf{x}}(k)$, affects several aspects of the system. The first obvious effect is on the dynamics of the resulting system in terms of stability, response speed, and damping. A secondary consideration, contradictory to the first, is the minimization of the energy expended during stair climbing. A balance of these two is achieved by selecting a damped system on the order of $\zeta = 0.7$ without affecting the natural frequency of the system significantly [27]. The effect of the controller design on the response of the system has also been iterated both in simulation and experimentally in order to refine the design.

V. Experimental Results

A. Implementation details

The estimation and control algorithms described in the preceding sections have been implemented on an iRobot Packbot tracked vehicle. The robot, shown in Fig. 1, is equipped with two retractable small arms, that are used as extensions of the tracks to facilitate climbing the first step of the stairs. After the initial alignment to the stairs (described in detail later in this section) is complete, the robot positions its arms at an angle of 60° from the ground, and starts approaching the stairs. Once the robot starts ascending, the arms are extended forward, to maximize traction. The two different positions of the robot's arms can be seen in Figs. 1 and 2.

The proprioceptive measurements in our implementation are provided by an Inertial Science ISIS IMU, operating at 100Hz. A Pointgrey Firefly camera is used, recording grayscale images at a rate of 15Hz, with a resolution of 640×480 pixels. The algorithms have been implemented in C++, and run in real time on a Pentium-3 onboard computer (800 MHz CPU, 256MB RAM) operating under Linux. The most computationally expensive procedure of the algorithm is the detection of the lines in the images, which requires approximately 60msec of processing time per image. The time necessary for propagating the state and the covariance is approximately 0.2msec, while the time needed for covariance update is approximately 3msec in the worst case (the actual update processing time depends on the number of detected lines).

Both sensors (gyroscope and camera) have been calibrated. Intrinsic camera calibration has been performed by application of Zhang's method [29], to estimate the linear parameters of the perspective model and the nonlinear distortion parameters. Using the resulting calibration, the pixel coordinates of image points can be transformed to the normalized image plane by employing the inverse model of Heikkila et al. [30]. The rotation between the camera and robot frames is known from the engineering drawings of the robot. The gyroscope calibration consists of determining the continuous-time standard deviation of the noise processes $\mathbf{n_r}$ and $\mathbf{n_w}$, which have been estimated as $\sigma_{r_c} = 6.3 \times 10^{-5}$(rad/sec)/$\sqrt{\text{Hz}}$, and $\sigma_{w_c} = 8 \times 10^{-6}$(rad/sec^2)/$\sqrt{\text{Hz}}$.

At the beginning of every run up the stairs, the state vector and its covariance must be initialized. An initial estimate for the gyroscopes' biases and their variance is produced by computing the sample mean and sample variance of gyroscope measurements, recorded while the robot remains static for 5sec. In order to initialize the attitude, we consider the ground at the bottom of the stairs approximately horizontal[6], and thus the only remaining unknown variable is the robot's rotation about the z-axis (yaw). This is estimated using the algorithm presented in [13], from the projections of lines in the image. If the robot is not initially aligned with the global coordinate frame, it rotates until the angle between the robot and global frames is smaller than a threshold (equal to 5° in our implementation).

The robot's attitude is initialized using the estimate for the robot's yaw after the initial alignment, and assuming zero rotation about the global x- and y-axes. The standard deviation of the initial attitude errors is set to 0.66° for the roll and pitch errors, and 2° for the yaw error. These values correspond to $\pm 3\sigma$ error intervals of $(-2^\circ, 2^\circ)$ for the roll and pitch errors, and $(-6^\circ, 6^\circ)$ for the yaw (cf. Fig 11). The relatively

[6] Alternatively, the roll and pitch angles can be determined from the values of the accelerometers of the IMU, or from an inclinometer, in case the robot is on uneven terrain.

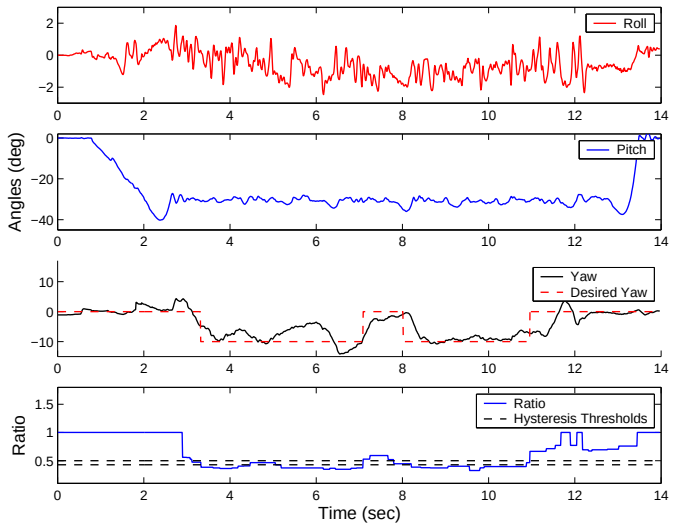


Fig. 9. The time evolution of the estimates for (i) the robot's attitude angles (top three plots) and (ii) the ratio of distances d_L/d_R (bottom plot).

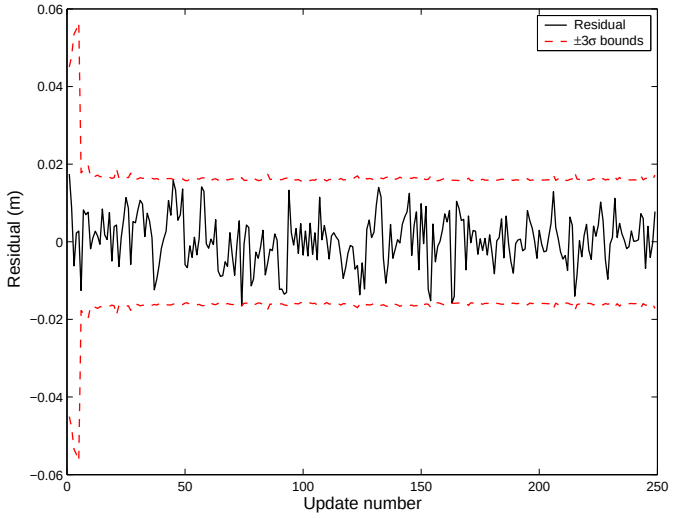


Fig. 10. The residuals of all lines that passed the Mahalanobis test, compared to the $\pm 3\sigma$ of their distribution.

large initial standard deviation for the yaw is chosen so as to allow for correcting potentially large errors in the initialization, which may result if the robot is too close to the stairs (and thus visibility is limited), or if spurious lines exist in the image.

As soon as the robot reaches the top of the stairs, it has to immediately detect this, stop, and switch to a different "behavior" (possibly searching for the next flight of stairs to climb [13]). Failure to do so may result in collisions and equipment damage. Since the latency of the EKF attitude estimates is very low (approximately 0.2msec), we have decided to employ these in order to detect the robot reaching the top of the stairs. In particular, when the robot's pitch in 10 consecutive time-steps (corresponding to a time interval of 0.1sec) is smaller than 5° in absolute value, the robot stops.

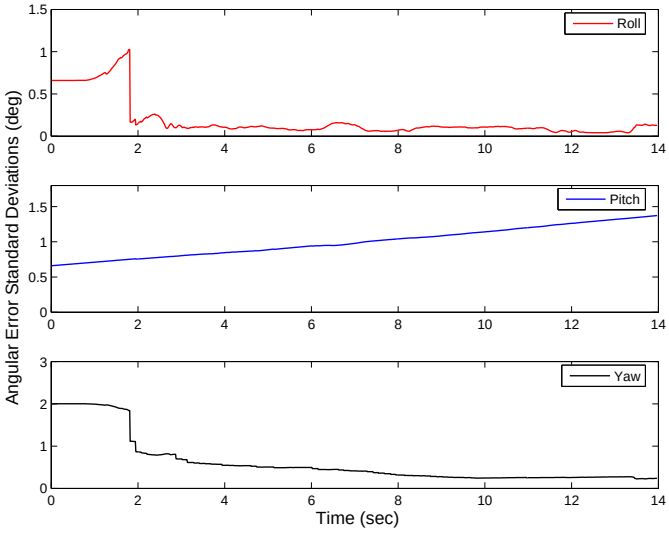


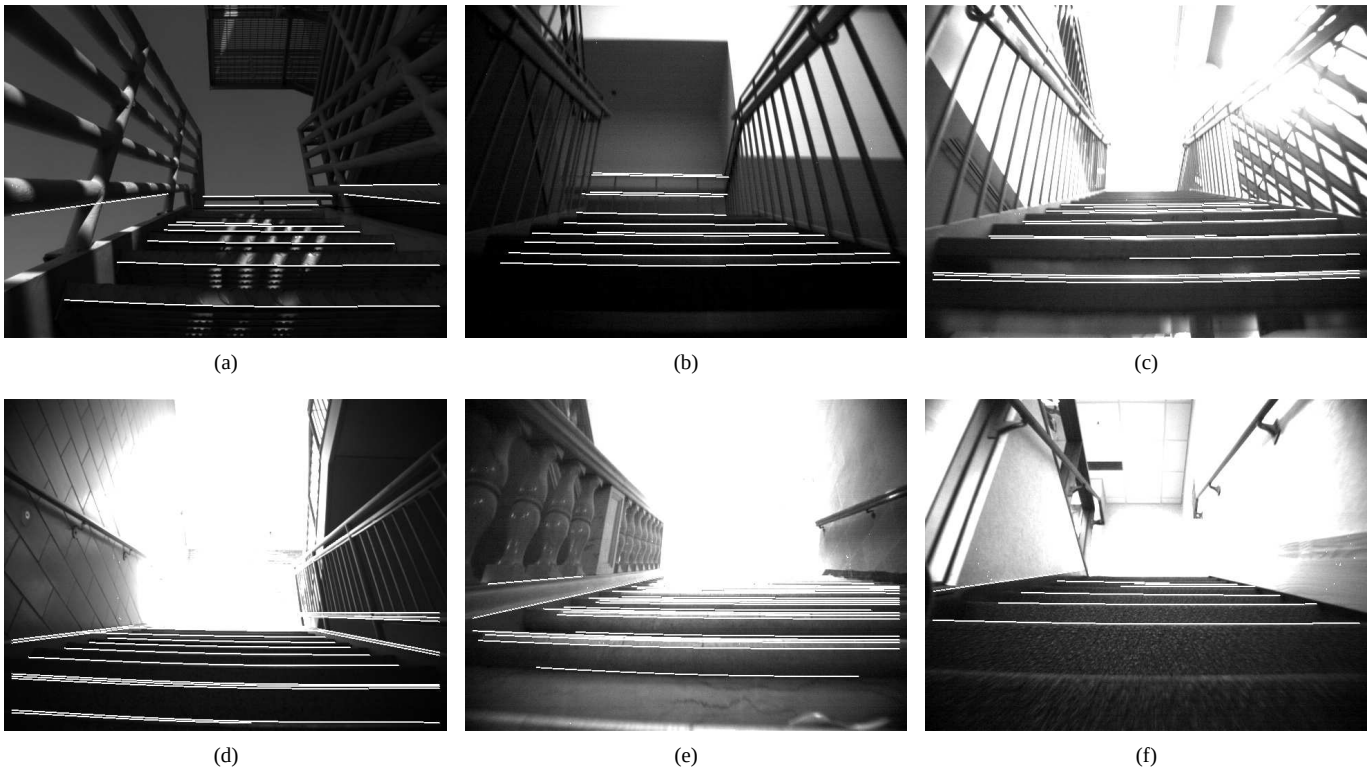
Fig. 11. The time evolution of the standard deviations for the angular errors.

B. Results

A large number of tests has been carried out to examine the performance of the proposed stair-climbing scheme, and we hereafter present representative results from one of the experimental runs. In Fig. 9, the estimated Euler X-Y-Z angles (roll-pitch-yaw) representing the robot's attitude, and the estimated distance ratio d_L/d_R are plotted. Note that the Euler angles are not directly estimated by the filter, in which a quaternion representation of rotation is used. They are presented in the figure to facilitate visualization, since plotting the time evolution of the quaternion elements does not provide an intuitive understanding of the robot's attitude. The robot's yaw angle is also compared with the reference angle θ_r determined by the centering controller.

In this run, the robot completed climbing the first step at approximately $t = 2.7$sec, and shortly after, the first reliable distance ratio became available. The robot correctly determined that it was positioned too close to the left wall, and the centering controller commanded the robot to head at an angle of 10° to the right (cf. Fig. 9). At approximately $t = 7$sec the robot entered the center zone, and therefore the reference angle of the heading controller became $\theta_r = 0^\circ$. However, due to slippage, the robot again moved to the left "non-safe zone" after 2sec (cf. Fig. 7), and the reference angle was set to -10° once again. At approximately $t = 11$sec, the robot's distance ratio crossed the threshold $\delta_c = 4/7$, and the robot remained in the center zone until it reached the top of the stairs, at approximately $t = 13$sec.

In Fig. 10, we plot the residuals of the line measurements, computed by Eq. (38), for all the lines that passed the gating test (Eq. (45)) during the run. These residuals are compared to the $\pm 3\sigma$ bounds corresponding to the diagonal elements of $\mathbf{S}$ (Eq. (47)). We observe that no noticeable bias is present, which indicates that the estimator is consistent, and that the employed sensor noise models are sufficiently accurate. The plots in Fig. 11 show the standard deviation of the angular errors. The plotted lines represent the square roots of the



(a) (b) (c)

(d) (e) (f)

Fig. 12. (a) Location: Tampa Police and Fire Training Academy tower, Tampa, FL, material: metal, slope: 35°, illumination: daylight (b) Location: CS&E Department 5th floor, University of Minnesota, material: plastic/carpet, slope: 28°, illumination: poor indoor lighting (c) Location: CS&E Department study commons, University of Minnesota, material: metal, slope: 30°, illumination: indoor lighting (d) Location: Columbia Heights Central Middle School, Minneapolis, MN, material: linoleum, slope: 30°, illumination: heavily back-lit (window on top of stairs) (e) Location: Walter Library lobby, University of Minnesota, material: marble, slope: 25°, illumination: indoor lighting (e) Location: Digital Technology Center, University of Minnesota, material: carpet, slope: 33°, illumination: indoor ambient daylight.

diagonal elements of the state covariance matrix corresponding to the attitude. From this figure, it becomes clear that the pitch is unobservable, as the variance of the errors around the robot's y axis monotonically increases. Contrary to that, the variance of the errors in the roll and yaw remains bounded, indicating that these degrees of freedom of the attitude are observable. These results corroborate the theoretical analysis of observability, presented in Appendix III.

Although we are not able to obtain ground truth attitude information for the entire duration of this experiment, we observe that the robot's pitch and roll angles at the top of the stairs are equal to 0.4° and 0.1°, respectively. In all our experimental runs, we have observed that the roll and pitch at the top of the stairs is consistently smaller than 1°, in absolute value. Comparing these results with the estimated standard deviations of the angular errors (equal to 0.15° for roll and 1.4° for pitch in this run), and taking into account the inaccuracies in the construction of the stairs, indicates that the covariance estimates accurately describe the uncertainty in the robot's attitude, and thus are consistent.

As shown in Fig. 9, the heading controller is able to reduce the error between the actual vehicle direction and that dictated by the centering controller to within roughly 5°. The variations from the nominal heading direction are due to disturbances in the system, caused by the dynamics of the interaction between the vehicle tracks and the stair steps, which are very difficult, if not impossible, to model. In contrast, the errors in the yaw estimates provided by the EKF become smaller than 1° after only a few seconds (cf. Fig. 11). Thus, the estimation errors are significantly smaller than the errors in the vehicle's commanded heading direction. This is actually the main reason for selecting sensor- instead of dynamic modeling when designing the estimator for this task. Similar cases have previously appeared in the literature (e.g., [15]) where even in the case of an orbiting satellite whose external disturbances are minimal, efforts to incorporate the vehicle dynamics in the design of the state estimator have not resulted in increased accuracy. On the contrary, the non-linear dynamics often have a negative impact on the performance of the estimator, as these introduce high-frequency components and biases that increase the errors in the state estimates [17].

We note at this point that the results presented in this section, that pertain to a single run of the robot up the stairs, are typical of the algorithm's performance. Averaging over all our recorded runs, the rms value of the deviation of the robot's heading from the commanded direction was equal to 3.54°, while the average value of the normalized distance ratio, $\delta = \min(d_L/d_R, d_R/d_L)$, was equal to 0.62. These values are computed using the estimates for the robot's attitude and for the ratio of distances d_L/d_R, as no ground truth is available. This performance has been determined, through extensive experimental validation, to be sufficient for the purposes of autonomous stair climbing.

C. Reliability

One of the primary concerns during the development of the stair-climbing algorithm is the algorithm's robustness to variations in environmental factors (e.g., illumination conditions, slope and appearance of stairs, slippage characteristics due to the surface material of the stairs, to name a few). We have conducted over 300 tests on different types of stairs, for example stairs covered by marble, metal, linoleum, and carpet, both indoors and outdoors, during different times of the day, and with slopes varying from 25° to 35°. It is worth noting that the algorithm has been successfully demonstrated at the NSF Industry/University Cooperative Research Center (I/U CRC) on Safety, Security, and Rescue Research (SSRRC) Spring 2005 Symposium in Tampa, FL, as well as at several community and industry outreach activities of the Digital Technology Center of the University of Minnesota. Example images from some of the tests we have performed are shown in Fig. 12. We note that camera gain calibration is performed adaptively based on the image intensity only in the part of the image where lines are detected. This often results in saturation in other parts of the image, especially when the stairs are less well-lit than the background. This approach, however, facilitates edge detection by increasing contrast in the areas of interest.

In our tests, we have consistently observed that orientation estimation is very accurate and robust. We attribute this to the high accuracy of the gyroscopes, and the effective outlier rejection (cf. Eq. (45)). The algorithm was able to correctly estimate the robot's heading in *all* the tests we performed. The only mode of failure that we have observed in our experiments is erroneous estimation of the ratio of distances to the left and right boundaries of the stairs. This only occurred in badly-lit indoor environments, when the surface of the stairs is covered by dark-colored material. In these cases, the endpoints of the stairs cannot always be reliably detected, thus sometimes resulting in the robot coming in contact with the wall or railing. This type of failure occurred in less than 10% of the cases where the robot attempted to climb dark and badly-lit stairs, and we believe that by placing a small light source on the robot, this problem can be eliminated.

VI. Conclusions

In this paper, we have presented an algorithm for autonomous stair climbing with a tracked vehicle. Through extensive experimentation, we have verified that this task can be accurately and reliably performed by a robot that receives and processes data from only two sensors: (i) the rotational velocity measurements provided by a 3-axial gyroscope, and (ii) the line parameters estimated from the stair-edges' projections on a camera image. Specifically, we have designed an EKF estimator that fuses these measurements and computes precise attitude estimates at a high rate. Additionally, we have described the process we employ for estimating the robot's relative distance to the stair ends, from the stair-edge measurements. This information is utilized by a centering controller that modifies the vehicle's heading direction every time the robot approaches the staircase boundaries. Finally, we have designed a state-feedback heading controller, based on the dynamics of the vehicle, that computes the required rotational velocities of the robot in order to steer the vehicle in the heading direction dictated by the centering controller. Contrary to previous approaches, our algorithm offers a tight integration of inertial and visual information, and can be applied on different robot models and stair types.

At this point we should note that the algorithm described in this paper, relies on the assumption that all stair edges are parallel, straight lines. Extending the algorithm to work in more general stairways, such as spiral staircases, is a possible direction of future research. Furthermore, we are currently investigating means to improve the robustness of estimating the ratio of the distances to the left and right stair boundaries. In the near future, we are planning to complement our existing algorithm with procedures for autonomous stair descent and automated search for stairs.

Acknowledgments

This work was supported by the University of Minnesota (DTC), the Jet Propulsion Laboratory (Grant No. 1251073, 1260245, 1263201), and the National Science Foundation (ITR-0324864, MRI-0420836). The authors would like to thank Joel Hesch, Le Vong Lo, Faraz Mirzaei, Kyle Smith, and Thor Andreas Tangen for their invaluable support during hardware/software development and experimental testing and validation. The authors would further like to thank the anonymous reviewers, who helped improve the quality of this paper through their insightful comments.

Appendix I
Discrete-Time Model

The discrete-time state transition matrix $\mathbf{\Phi}_k$ is a block matrix with the following structure:

$$\mathbf{\Phi}_k = \begin{bmatrix} \mathbf{\Theta} & \mathbf{\Psi} \\ \mathbf{0}_{3\times 3} & \mathbf{I}_{3\times 3} \end{bmatrix} \tag{69}$$

The matrices $\mathbf{\Theta}$ and $\mathbf{\Psi}$ can be computed as

$$\begin{aligned} \mathbf{\Theta} = \mathbf{I}_{3\times 3} &- \frac{1}{|\hat{\boldsymbol{\omega}}|}\sin\left(|\hat{\boldsymbol{\omega}}|\Delta t\right)\lfloor\hat{\boldsymbol{\omega}}\times\rfloor \\ &+ \frac{1}{|\hat{\boldsymbol{\omega}}|^2}\left(1-\cos(|\hat{\boldsymbol{\omega}}|\Delta t)\right)\lfloor\hat{\boldsymbol{\omega}}\times\rfloor^2 \end{aligned} \tag{70}$$

$$\begin{aligned} \mathbf{\Psi} = -\mathbf{I}_{3\times 3}\Delta t &+ \frac{1}{|\hat{\boldsymbol{\omega}}|^2}\left(1-\cos(|\hat{\boldsymbol{\omega}}|\Delta t)\right)\lfloor\hat{\boldsymbol{\omega}}\times\rfloor \\ &- \frac{1}{|\hat{\boldsymbol{\omega}}|^3}\left(|\hat{\boldsymbol{\omega}}|\Delta t-\sin(|\hat{\boldsymbol{\omega}}|\Delta t)\right)\lfloor\hat{\boldsymbol{\omega}}\times\rfloor^2 \end{aligned} \tag{71}$$

When $|\hat{\boldsymbol{\omega}}|$ is small, both of the above expressions will lead to numerical instability. By taking the limit and applying L'Hôpital's rule, we arrive at

$$\lim_{|\hat{\boldsymbol{\omega}}|\to 0}\mathbf{\Theta} = \mathbf{I}_{3\times 3} - \Delta t\lfloor\hat{\boldsymbol{\omega}}\times\rfloor + \frac{\Delta t^2}{2}\lfloor\hat{\boldsymbol{\omega}}\times\rfloor^2 \tag{72}$$

$$\lim_{|\hat{\boldsymbol{\omega}}|\to 0}\mathbf{\Psi} = -\mathbf{I}_{3\times 3}\Delta t + \frac{\Delta t^2}{2}\lfloor\hat{\boldsymbol{\omega}}\times\rfloor - \frac{\Delta t^3}{6}\lfloor\hat{\boldsymbol{\omega}}\times\rfloor^2 \tag{73}$$

The discrete-time noise covariance matrix $\mathbf{Q}_d$ has the following structure

$$\mathbf{Q}_d = \begin{bmatrix} \mathbf{Q}_{11} & \mathbf{Q}_{12} \\ \mathbf{Q}_{12}^{\mathrm{T}} & \mathbf{Q}_{22} \end{bmatrix} \tag{74}$$

and the elements follow after considerable algebra as

$$\mathbf{Q}_{11} = \sigma_r^2 \Delta t \cdot \mathbf{I}_{3\times 3} + \sigma_w^2 \cdot \Bigg(\mathbf{I}_{3\times 3} \frac{\Delta t^3}{3} + \frac{\frac{(|\hat{\boldsymbol{\omega}}|\Delta t)^3}{3} + 2\sin(|\hat{\boldsymbol{\omega}}|\Delta t) - 2|\hat{\boldsymbol{\omega}}|\Delta t}{|\hat{\boldsymbol{\omega}}|^5} \cdot \lfloor \hat{\boldsymbol{\omega}} \times \rfloor^2 \Bigg) \tag{75}$$

$$\mathbf{Q}_{12} = -\sigma_w^2 \cdot \Bigg(\mathbf{I}_{3\times 3} \frac{\Delta t^2}{2} - \frac{|\hat{\boldsymbol{\omega}}|\Delta t - \sin(|\hat{\boldsymbol{\omega}}|\Delta t)}{|\hat{\boldsymbol{\omega}}|^3} \cdot \lfloor \hat{\boldsymbol{\omega}} \times \rfloor + \frac{\frac{(|\hat{\boldsymbol{\omega}}|\Delta t)^2}{2} + \cos(|\hat{\boldsymbol{\omega}}|\Delta t) - 1}{|\hat{\boldsymbol{\omega}}|^4} \cdot \lfloor \hat{\boldsymbol{\omega}} \times \rfloor^2 \Bigg) \tag{76}$$

$$\mathbf{Q}_{22} = \sigma_w^2 \Delta t \cdot \mathbf{I}_{3\times 3} \tag{77}$$

As in the case of the state transition matrix, we can derive the form for small $|\hat{\boldsymbol{\omega}}|$ by taking the limit and applying L'Hôpital's rule

$$\lim_{|\hat{\boldsymbol{\omega}}|\to 0} \mathbf{Q}_{11} = \sigma_r^2 \Delta t \cdot \mathbf{I}_{3\times 3} + 2\sigma_w^2 \left(\mathbf{I}_{3\times 3} \frac{\Delta t^3}{3!} + \frac{\Delta t^5}{5!} \cdot \lfloor \hat{\boldsymbol{\omega}} \times \rfloor^2 \right)$$

$$\lim_{|\hat{\boldsymbol{\omega}}|\to 0} \mathbf{Q}_{12} = -\sigma_w^2 \cdot \left(\mathbf{I}_{3\times 3} \frac{\Delta t^2}{2!} - \frac{\Delta t^3}{3!} \cdot \lfloor \hat{\boldsymbol{\omega}} \times \rfloor + \frac{\Delta t^4}{4!} \cdot \lfloor \hat{\boldsymbol{\omega}} \times \rfloor^2 \right)$$

For a detailed derivation of the above expressions, the interested reader is referred to [20].

APPENDIX II
LINE EXTRACTION

The images recorded by the robot's onboard camera are processed to detect the projections of the stair edges. The detected lines are employed (i) for updating the robot's attitude estimate, and (ii) for estimating the ratio of the robot's distances to the left and right boundaries of the stairs. In the following, we outline the steps of the straight-line detection algorithm.

A. Edge detection

The first step in the processing of each image involves application of Canny's edge detection operator [31]. In order to achieve invariance of the edge detection procedure to illumination changes, as well as to the effects of blurring, that is caused by the robot's rapid orientation changes, the thresholds in the Canny algorithm are selected adaptively. In particular, the standard deviation, σ_G, of the image gradient along the vertical image direction is computed, and the cutoff-values in Canny's hysteresis-based edge thresholding are selected as $(\sigma_G, \sigma_G/4)$.

B. Straight-line detection

The output of the edge detection process is a set of edge segments. Given the normalized coordinates $p_i = (u_i, v_i)$, $i = 1 \ldots N$ of the points in the j-th edge segment, total least-squares line-fitting is performed to obtain an estimate of the best straight-line fit. Lines are parameterized using the polar representation (cf. Eq. (34)), and the line parameters (ϕ_j, ρ_j) are determined by minimizing the weighted sum of squared distances of all the points p_i to the line (cf. Eq. (35)):

$$\begin{aligned}(\phi_j, \rho_j) &= \arg\min_{\phi,\rho} J(\phi, \rho) \\ &= \arg\min_{\phi,\rho} \sum_{i=1}^{N} \frac{1}{\sigma^2} \left(u_i \cos\phi + v_i \sin\phi - \rho \right)^2 \end{aligned} \tag{78}$$

where σ is the standard deviation of the errors in the image coordinates of the detected edge points.

For each line, the covariance matrix of the line parameters is computed, and denoted as

$$\mathbf{P}_{\boldsymbol{\ell}_j} = \begin{bmatrix} \sigma_{\phi_j}^2 & \mathrm{corr}(\phi_j, \rho_j) \\ \mathrm{corr}(\phi_j, \rho_j) & \sigma_{\rho_j}^2 \end{bmatrix} \tag{79}$$

where $\sigma_{\phi_j}^2$ is the variance of the line's orientation, $\sigma_{\rho_j}^2$ is the variance of the line's distance from the origin of the image coordinate frame, and $\mathrm{corr}(\phi_j, \rho_j)$ is the correlation between the line orientation and distance.

In order to discard all segments that do not correspond to straight lines, we perform a χ^2 compatibility test. Specifically, the weighted sum of the squared distances of all points p_i to the line $\boldsymbol{\ell}_j$, $J(\phi_j, \rho_j)$, is a random variable, distributed according to χ^2_{N-2}. In order to filter out edge segments that do not correspond to straight lines, we discard edges with values for $J(\phi_j, \rho_j)$ exceeding a threshold equal to the 99-percentile of the χ^2_{N-2} distribution.

Once all the straight lines in the image have been detected, we examine whether some of the detected lines correspond to the same physical line. For this purpose, all lines are examined in pairs, and if the difference in the lines' parameters is small, total least-squares line-fitting is performed using the points that belong to both lines. If the resulting line satisfies the aforementioned χ^2 criterion, then the two lines are merged. This process is applied recursively, until no more lines can be merged.

In the EKF update of the robot's attitude, the covariance matrix of the line equation vector, $\boldsymbol{\ell}_j$, is necessary (cf. Section III-A.9). This is computed as:

$$\begin{aligned}\mathbf{R}_j &= \left(\nabla_{[\phi_j \ \rho_j]^T} \boldsymbol{\ell}_j \right) \mathbf{P}_{\boldsymbol{\ell}_j} \left(\nabla_{[\phi_j \ \rho_j]^T} \boldsymbol{\ell}_j \right)^T \\ &= \begin{bmatrix} -\sin\phi_j & 0 \\ \cos\phi_j & 0 \\ 0 & -1 \end{bmatrix} \mathbf{P}_{\boldsymbol{\ell}_j} \begin{bmatrix} -\sin\phi_j & \cos\phi_j & 0 \\ 0 & 0 & -1 \end{bmatrix} \end{aligned} \tag{80}$$

APPENDIX III
OBSERVABILITY ANALYSIS

In order to analyze the stochastic observability of the proposed attitude filter, we will examine a slightly simplified

system that neglects the gyroscope bias.[7] Similarly to Eq. (46) in Section III-A.9, we can then write the residual for all line measurements as

$$\mathbf{r} = \mathbf{H}_q \tilde{\mathbf{x}}' + \boldsymbol{\eta} \tag{81}$$

In this case, however, $\tilde{\mathbf{x}}' = \delta\boldsymbol{\theta}$, so that $\mathbf{H}_q$ is a matrix with rows

$$\mathbf{H}_{q_j} = \mathbf{e}_i^T \mathbf{C}^T(\bar{q}) \lfloor \mathbf{C}^T({}^C_L\bar{q}) \boldsymbol{\ell}_j \times \rfloor \tag{82}$$

The noise term $\boldsymbol{\eta} = \boldsymbol{\Gamma}\mathbf{n}$ is Gaussian with zero mean and covariance

$$\mathrm{cov}(\boldsymbol{\eta}) = \boldsymbol{\Gamma}\mathbf{R}\boldsymbol{\Gamma}^T \tag{83}$$

As before, $\boldsymbol{\Gamma}$ and $\mathbf{R}$ are block-diagonal matrices, with diagonal elements

$$\boldsymbol{\Gamma}_j = \mathbf{e}_i^T \mathbf{C}^T(\bar{q}) \mathbf{C}^T({}^C_L\bar{q}) \tag{84}$$

and $\mathbf{R}_j$ as in Eq. (80).

From the block-diagonal structure of $\boldsymbol{\Gamma}$ and $\mathbf{R}$, we see that $\mathrm{cov}(\boldsymbol{\eta})$ is a diagonal matrix with positive, scalar diagonal elements

$$\mathrm{cov}(\boldsymbol{\eta})_{jj} = \boldsymbol{\Gamma}_j \mathbf{R}_j \boldsymbol{\Gamma}_j^T \tag{85}$$

Recalling the definition of $\mathbf{R}_j$ from Eq. (80), we see that it is of rank 2, assuming that $\mathbf{P}_{\boldsymbol{\ell}_j}$ is of full rank. Moreover,

$$\mathrm{Null}(\mathbf{R}_j) = \gamma \begin{bmatrix} c\phi_j \\ s\phi_j \\ 0 \end{bmatrix}, \quad \gamma \in \mathbb{R} \tag{86}$$

In order for $\mathrm{cov}(\boldsymbol{\eta})_{jj}$ to be zero, $\boldsymbol{\Gamma}_j^T = \mathbf{C}({}^C_L\bar{q})\mathbf{C}(\bar{q}){}^G\mathbf{e}_i = {}^C\mathbf{e}_i$ must lie in this nullspace. Assuming that this is the case, we proceed by applying the measurement constraint, Eq. (37)

$${}^C\mathbf{e}_i^T \boldsymbol{\ell}_j = 0 \quad \Rightarrow \quad \gamma \begin{bmatrix} c\phi_j & s\phi_j & 0 \end{bmatrix} \begin{bmatrix} c\phi_j \\ s\phi_j \\ -\rho_j \end{bmatrix} = 0 \tag{87}$$

$$\Rightarrow \gamma = 0 \quad \Rightarrow \quad {}^C\mathbf{e}_i = \mathbf{0} \tag{88}$$

Obviously, ${}^C\mathbf{e}_i = \mathbf{0}$ is impossible, since by definition $||\mathbf{e}_i|| = 1$, which completes the proof by contradiction. We can therefore conclude, that all diagonal elements of $\boldsymbol{\Gamma}\mathbf{R}\boldsymbol{\Gamma}^T$ are different from zero, so that $\mathrm{cov}(\boldsymbol{\eta})$ is always invertible.

Stochastic observability requires that there exist positive constants α, β with $\alpha < \beta < \infty$ and a positive integer N such that, for all $\nu \geq N$,

$$\alpha \mathbf{I}_{3\times 3} \leq \sum_{\mu=\nu-N+1}^{\nu} \boldsymbol{\Phi}_q^T(t_\mu, t_\nu) \mathbf{H}_q^T(t_\mu) \mathrm{cov}(\boldsymbol{\eta})^{-1}(t_\mu) \mathbf{H}_q(t_\mu) \boldsymbol{\Phi}_q(t_\mu, t_\nu) \leq \beta \mathbf{I}_{3\times 3} \tag{89}$$

or, in other words, that the inner sum of matrices is of full rank [17].

Note that Eq. (70) implies that $\boldsymbol{\Phi}_q = \boldsymbol{\Theta}$ is the rotational matrix

$$\boldsymbol{\Phi}_q(t_\mu, t_\nu) = \mathbf{C}({}^{L_\mu}_{L_\nu}\bar{q}) \tag{90}$$

[7] Intuitively, if the attitude is observable, the bias is also observable, as can be verified for a simple 1-D example.

Pre- and post-multiplication of Eq. (89) with $\mathbf{C}^T({}^{L_\nu}_G\bar{q})$ and $\mathbf{C}({}^{L_\nu}_G\bar{q})$ respectively, changes the inner expression to

$$\sum_{\mu=\nu-N+1}^{\nu} \mathbf{C}({}^G_{L_\mu}\bar{q}) \mathbf{H}_q^T(t_\mu) \mathrm{cov}(\boldsymbol{\eta})^{-1}(t_\mu) \mathbf{H}_q(t_\mu) \mathbf{C}({}^{L_\mu}_G\bar{q}) \tag{91}$$

Due to the particular block-structure of $\mathbf{H}_q$ and $\mathrm{cov}(\boldsymbol{\eta})$, this can be decomposed as

$$\sum_{\mu=\nu-N+1}^{\nu} \sum_{j=1}^{M} \mathrm{cov}(\boldsymbol{\eta})_{jj}^{-1} \mathbf{C}({}^G_{L_\mu}\bar{q}) \mathbf{H}_{q_j}^T(t_\mu) \mathbf{H}_{q_j}(t_\mu) \mathbf{C}({}^{L_\mu}_G\bar{q})$$

which is a sum of outer product matrices, since ${}^G\mathbf{H}_{q_j}^T = \mathbf{C}({}^G_{L_\mu}\bar{q})\mathbf{H}_{q_j}^T$ is of dimension 3×1. The rank of the observability matrix is therefore equal to the number of linearly independent vectors ${}^G\mathbf{H}_{q_j}^T$. Algebraic transformation of ${}^G\mathbf{H}_{q_j}^T$ shows that this vector is the cross product of the line vector $\boldsymbol{\ell}_j$ and the unit vector $\mathbf{e}_i$, both expressed in global coordinates:

$$\begin{aligned} {}^G\mathbf{H}_{q_j}^T &= \mathbf{C}^T(\bar{q}) \big(\mathbf{e}_i^T \mathbf{C}^T(\bar{q}) \lfloor \mathbf{C}^T({}^C_L\bar{q}) \boldsymbol{\ell}_j \times \rfloor \big)^T \\ &= \big(\mathbf{e}_i^T \lfloor \mathbf{C}^T(\bar{q}) \mathbf{C}^T({}^C_L\bar{q}) \boldsymbol{\ell}_j \times \rfloor \big)^T \\ &= - \lfloor {}^G\boldsymbol{\ell}_j \times \rfloor {}^G\mathbf{e}_i \\ &= {}^G\mathbf{e}_i \times {}^G\boldsymbol{\ell}_j \end{aligned}$$

This implies that all $\mathbf{H}_{q_j}^T$ are orthogonal to $\mathbf{e}_i$ and are thus confined to one plane (i.e., do not span the 3D space). We can therefore conclude that if we observe only edges parallel to one unit vector (as is the case for stair edges parallel to the global y-axis), the observability matrix will be at most of rank 2. Note that this does not change *regardless of the robot's trajectory*. The rotation about the observed unit vector $\mathbf{e}_i$, in our case the pitch angle, will remain unobservable, as corroborated by Fig. 11.

However, in the general case where at least two linearly independent unit vectors $\mathbf{v}_i$, $i = 1, 2$ are observed three times, the attitude becomes fully observable. This is equivalent to the matrix $\mathbf{H}_q^T = \begin{bmatrix} \mathbf{H}_{q_1}^T & \mathbf{H}_{q_2}^T & \mathbf{H}_{q_3}^T \end{bmatrix}$ having full rank, where $\mathbf{H}_{q_j}^T = \mathbf{v}_i \times \boldsymbol{\ell}_j$, $i = 1$ or 2, $j = 1, 2, 3$, and $\mathbf{v}_i$ and $\boldsymbol{\ell}_j$ are expressed with respect to the same coordinate frame. Depending on the vectors $\mathbf{v}_i$ and the observed lines $\boldsymbol{\ell}_j$, there can arise several singular cases, of which the following is of particular interest:

- $\mathbf{v}_i = \mathbf{v}$: If we only observe one unit vector, the matrix will be at best of rank 2. As discussed previously, this is the case when the robot observes only stair edges, i.e., $\mathbf{v}_i = \mathbf{e}_2$.
- $\boldsymbol{\ell}_j = \boldsymbol{\ell}$: This condition requires that all unit vectors project along the same line on the image (i.e., only one line direction is observed in the image data).

REFERENCES

[1] J. Martens and W. Newman, "Stabilization of a mobile robot climbing stairs," in *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, vol. 3, San Diego, CA, May 1994, pp. 2501–2507.

[2] iRobot Corporation. PackBot Homepage. www.packbot.com.

[3] REMOTEC, INC. ANDROS Remote Vehicle Homepage. www.remotec-andros.com.

[4] M. Lawn and T. Ishimatsu, "Modeling of a stair-climbing wheelchair mechanism with high single-step capability," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 11, no. 3, pp. 323–332, Sept. 2003.

[5] G. Figliolini and M. Ceccarelli, "Climbing stairs with EP-WAR2 biped robot," in *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, vol. 4, Seoul, Korea, 2001, pp. 4116–4121.

[6] A. Albert, M. Suppa, and W. Gerth, "Detection of stair dimensions for the path planning of a bipedal robot," in *Proc. IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, vol. 2, Como, Italy, July 2001, pp. 1291–1296.

[7] J.-S. Gutmann, M. Fukuchi, and M. Fujita, "Stair climbing for humanoid robots using stereo vision," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, vol. 2, Sendai, Japan, 28 Sept.-2 Oct. 2004, pp. 1407–1413.

[8] K. Hirai, M. Hirose, Y. Haikawa, and T. Takenaka, "The development of Honda humanoid robot," in *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, vol. 2, Leuven, Belgium, May 1998, pp. 1321–1326.

[9] E. Z. Moore, D. Campbell, F. Grimminger, and M. Buehler, "Reliable stair climbing in the simple hexapod 'RHex'," in *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*, vol. 3, Washington, DC, May 2002, pp. 2222–2227.

[10] O. Matsumoto, S. Kajita, and K. Komoriya, "Flexible locomotion control of a self-contained biped leg-wheeled system," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, vol. 3, Lausanne, Switzerland, 30 Sept.-5 Oct. 2002, pp. 2599–2604.

[11] J. Liu, Y. Wang, S. Ma, and B. Li, "Analysis of stairs-climbing ability for a tracked reconfigurable modular robot," in *Proc. IEEE International Workshop on Safety, Security and Rescue Robotics*, Kobe, Japan, June 2005, pp. 53–38.

[12] S. Steplight, G. Egnal, S.-H. Jung, D. Walker, C. Taylor, and J. Ostrowski, "A mode-based sensor fusion approach to robotic stair-climbing," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, vol. 2, Takamatsu, Japan, Nov. 2000, pp. 1113–1118.

[13] Y. Xiong and L. Matthies, "Vision-guided autonomous stair climbing," in *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, vol. 2, San Francisco, CA, Apr. 2000, pp. 1842–1847.

[14] D. Helmick, S. I. Roumeliotis, M. McHenry, and L. Matthies, "Multi-sensor, high speed autonomous stair climbing," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, vol. 1, Lausanne, Switzerland, 30 Sept.-5 Oct. 2002, pp. 733–742.

[15] E. J. Lefferts and F. L. Markley, "Dynamics modeling for attitude determination," *AIAA Paper 76-1910*, Aug. 1976.

[16] S. I. Roumeliotis, G. S. Sukhatme, and G. A. Bekey, "Circumventing dynamic modeling: Evaluation of the error-state Kalman filter applied to mobile robot localization," in *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, vol. 2, Detroit, MI, May 10-15 1999, pp. 1656–1663.

[17] P. S. Maybeck, *Stochastic Models, Estimation and Control*. New York: Academic Press, 1979, vol. 1.

[18] M. D. Shuster, "A survey of attitude representations," *Journal of the Astronautical Sciences*, vol. 41, no. 4, pp. 439–517, Oct.–Dec. 1993.

[19] E. J. Lefferts, F. L. Markley, and M. D. Shuster, "Kalman filtering for spacecraft attitude estimation," *Journal of Guidance, Control, and Dynamics*, vol. 5, no. 5, pp. 417–429, Sept.-Oct. 1982.

[20] N. Trawny and S. I. Roumeliotis, "Indirect Kalman filter for 3D attitude estimation," University of Minnesota, Dept. of Comp. Sci. & Eng., Tech. Rep. 2005-002, Jan. 2005.

[21] W. G. Breckenridge, "Quaternions - Proposed Standard Conventions," JPL Interoffice Memorandum IOM 343-79-1199, 1999.

[22] J. J. Craig, *Introduction to robotics: mechanics and control*, 3rd ed. Upper Saddle River, New Jersey: Pearson Prentice Hall, 2005.

[23] J. R. Wertz, Ed., *Spacecraft Attitude Determination and Control*. Dordrecht; Boston: Kluwer Academic, 1978.

[24] S. I. Roumeliotis and J. W. Burdick, "Stochastic cloning: A generalized framework for processing relative state measurements," in *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, Washington, DC, 2002, pp. 1788–1795.

[25] J. Wong, *Theory of Ground Vehicles*, 3rd ed. Wiley-Interscience, 2001.

[26] M. Bekker, *Theory of land locomotion: The mechanics of vehicle mobility*. University of Michigan Press, 1956.

[27] G. Franklin, J. Powell, and M. Workman, *Digital Control of Dynamic Systems*. Prentice Hall, 1997.

[28] I. Hayashi, "Practical analysis of tracked vehicle steering depending on longitudinal track slippage," in *5th International Conference of the International Society for Terrain-Vehicle Systems*, Detroit-Houghton, MI, 1975.

[29] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 11, pp. 1330–1334, 2000.

[30] J. Heikkila and O. Silven, "A four-step camera calibration procedure with implicit image correction," in *Proceedings of the 1997 Conference on Computer Vision and Pattern Recognition (CVPR '97)*, Washington, DC, 1997, pp. 1106–1113.

[31] J. Canny, "A computational approach to edge detection," *IEEE Transacions on Pattern Analysis and Machine Intelligence*, vol. 8, no. 6, pp. 679–698, 1986.