Interval-Based Visual-LiDAR Sensor Fusion

Raphael Voges and Bernardo Wagner

***Abstract*—Since cameras and Light Detection and Ranging (LiDAR) sensors provide complementary information about the environment, it is beneficial for mobile robot localization to fuse their information by assigning distances measured by the LiDAR to visual features detected in the image. However, existing approaches neglect the uncertainty of the fused information or model it in an optimistic way (e.g. without taking extrinsic calibration errors into account). Since the actual distribution of errors during sensor fusion is often unknown, we assume to only know bounds (or intervals) enclosing the errors. Consequently, we propose to use interval analysis to propagate the error from the input sources to the fused information in a straightforward way. To show the applicability of our approach, we use the fused information for dead reckoning. Since interval analysis is used, the result of our approach are intervals that are guaranteed to enclose the robot's true pose. An evaluation using real data shows that we are indeed able to localize the robot in a guaranteed way. This enables us to detect faults of an established approach, which neglects the uncertainty of the fused information, in three out of ten cases.**

***Index Terms*—Sensor Fusion, Formal Methods in Robotics and Automation, Localization, Interval Analysis**

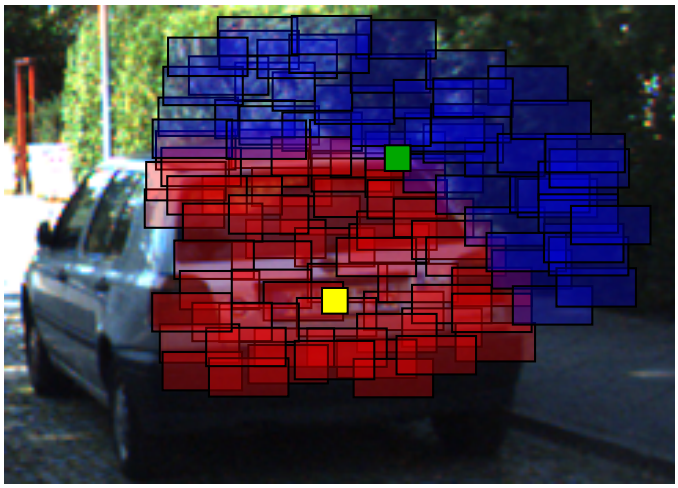


Fig. 1. Two image features and projected laser scan boxes that are colored by depth (red: close, blue: distant). Since the yellow image feature intersects only scan boxes of similar depths, we can accurately compute its depth. In contrast, the green image feature intersects scan boxes from both fore- and background, and thus its depth uncertainty will be large.

I. Introduction

In the case of missing GNSS information, mobile robots have to localize using different sensors such as cameras or LiDARs. While the camera provides rich features that can be reidentified, the depth (or distance) of features is hard to determine robustly and is "liable to drift over time" [1]. In contrast, the LiDAR is able to accurately measure distances, but does not provide easily reidentifiable features. To fuse information from both sensors, existing approaches assign distances measured by the LiDAR to individual image features. This results in 3D features that can be reidentified in consecutive images. However, most approaches do not propagate the sensor and the inter-sensor (e.g. the transformation between sensor coordinate systems) errors consistently during sensor fusion.

This leads to two problems. First, the accuracy of the fused 3D features cannot be assessed, and therefore these features cannot be weighted according to their reliability for robot localization. Moreover, the accuracy of further results based on these features cannot be determined either. Second, the transformation between the sensor coordinate systems, which is required to fuse information, is assumed to be known without error. However, this transformation is usually only known up to some accuracy since it needs to be determined beforehand during an extrinsic calibration using sensor data [2]. Depending on the error of the estimated transformation, scan points and image features corresponding to different objects might be fused, thereby creating incorrectly fused 3D features.

To overcome these problems, we propose to take all sensor and inter-sensor for sensor fusion of camera and LiDAR into account. However, the true error distribution of off-the-shelf sensors is generally unknown or depends on the environment. For example, the incidence angle and the reflectivity of the surface influence the distance measurement of a LiDAR [3]. Similarly, the accuracy with which image features can be reidentified in consecutive images depends, for example, on the image blur and the environment. Therefore, we propose to use interval analysis [4] to model all errors and introduce bounded error models for camera and LiDAR (cf. Section IV).

In contrast to stochastic models, the idea is to assume an unknown but bounded error for every input source which means that we assume to not know exact measurement values but only an interval that is guaranteed to contain the true value. Consequently, as Fig. 1 shows, image features and laser scan data are no longer points but boxes. When trying to find the distance of an image feature, modeling it as an interval allows us to dynamically determine all surrounding eligible laser scan points (namely all points intersecting the image feature box) instead of arbitrarily selecting the n closest points. Since these laser scan points are also modeled using intervals, we can compute an upper and lower bound for the depth of

Manuscript received October 14, 2020; Revised December 31, 2020; Accepted January 21, 2021.
This paper was recommended for publication by Editor E. Marchand upon evaluation of the Associate Editor and Reviewers' comments. This work was supported by the German Research Foundation (DFG) as part of the Research Training Group i.c.sens [RTG 2159].
The authors are with the Real Time Systems Group (RTS), Institute of Systems Engineering, Leibniz Universität Hannover, D-30167 Hannover, Germany. {voges,wagner}@rts.uni-hannover.de
Digital Object Identifier (DOI): see top of this page.

the feature, taking the "worst case", i.e. the maximum error, into account. Consequently, the uncertainty of the 3D feature results directly from the sensor and inter-sensor uncertainties. Section V introduces our approach.

Subsequently, we employ the fused 3D features to perform dead reckoning based on camera and LiDAR data (also known as visual-LiDAR odometry), i.e. incremental localization starting from an arbitrary position, of a mobile robot. An Inertial Measurement Unit (IMU) is used to initially constrain the rotation. In Section VI we formulate the dead reckoning problem as a Constraint Satisfaction Problem (CSP) and solve it using interval analysis tools. Under the assumption of correct error bounds, our approach allows us to compute an interval box which is guaranteed to contain the robot's true pose. Still, outliers may occur (e.g. due to incorrect feature matches), but can be taken into account. If the true pose is not enclosed by the interval box, we can trace the error directly back to the incorrectly assumed error bounds, since all computations are deterministic. Consequently, our approach is especially useful for safety-critical systems (e.g. autonomous driving) in which we want to guarantee results under given assumptions and, in case of an error, want to identify the wrong assumption leading to the error. Finally, we show the feasibility of our method using real data (cf. Section VII).

II. Related Work

There are two different ways to fuse information from camera and LiDAR. The simpler approach is to independently extract and track features from both LiDAR and image data before combining those features in a common Extended Kalman Filter (EKF) [5]. However, this approach still suffers from the drawbacks of the individual sensors since camera features still miss accurate distance information and laser scan points cannot be associated easily over time.

The second, more sophisticated, approach is to fuse sensor information directly by assigning distances to image features, which can be reidentified in consecutive images. This allows to use the benefits of both sensor modalities. Zhang et al. [6] find the depth for an image feature by determining the three closest laser points and interpolating the distance of these. Graeter et al. [7] fuse the information from LiDAR and camera in a similar fashion. They select a set of laser scan points around each image feature and determine its depth by fitting a plane to the set of scan points. However, both approaches neglect any uncertainties during this procedure. In contrast, Andert et al. [8] only find the closest laser point for each image feature and assign the distance of this point to the image feature. In addition, they determine the uncertainty of the fused feature by combining the individual uncertainties of the image feature and the laser scan point. While this error model may be adequate if laser scan points around the image feature have roughly the same distance (e.g. image features on planes perpendicular to the viewing direction), it can be overly optimistic in the case of image features in areas containing large distance gradients (e.g. image features on edges). Besides, inter-sensor errors are neglected and the exact error distribution of both camera and LiDAR must be known. However, this is usually not the case with off-the-shelf sensors.

III. Interval Analysis

This short introduction to interval analysis is based on [2], [4]. Interval analysis can be understood as the extension of classical real arithmetic operators to set theory. However, instead of using arbitrary sets, computations are performed on intervals, which are defined by a lower and an upper bound, $\underline{x}$ and $\overline{x}$ respectively. Consequently, an interval $[x] = [\underline{x}, \overline{x}]$ is defined as the set of all real numbers between those bounds. Such bounds are intuitively stated by us humans when talking about, for example, a distance that we know is accurate up to $\pm 5\,\mathrm{m}$. The implicit meaning is that we only know the true value x^* is guaranteed to be in the interval $[x]$, i.e. $x^* \in [x]$, but we cannot access the true value, and thus do not know which value in the interval is most likely. A vector of intervals is denoted as an interval box $[\mathbf{x}]$. We show an exemplary subtraction of two intervals:

$$[-4, 3] - [1, 5] = [-4 - 5, 3 - 1] = [-9, 2]. \tag{1}$$

All possible value combinations from both intervals are taken into account. Thus, this way of computing can also be seen as a worst-case consideration. Given that the specified error bounds are correct, interval-based approaches guarantee to enclose the true solution.

State estimation often requires to characterize the set (e.g. the sought position of a robot)

$$\mathbb{X} = \{\, \mathbf{x} \in \mathbb{R}^n \mid \mathbf{f}(\mathbf{x}) \in \mathbb{Y} \,\} = \mathbf{f}^{-1}(\mathbb{Y}), \tag{2}$$

where $\mathbb{Y} \subset \mathbb{R}^m$ (e.g. a measurement to a landmark) and $\mathbf{f}$ is a possibly nonlinear function $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^m$ (e.g. a measurement function describing the expected measurement given the robot's position). An exponential time algorithm (since a branch and bound strategy is employed) to compute $\mathbb{X}$ is the Set Inversion Via Interval Analysis (SIVIA) [4].

To avoid the exponential time requirement, the problem can also be formulated as a CSP. $\mathbf{f}$ can be understood as constraints on the variables $\mathbf{x}$. Then, given an initial search space, which can be arbitrarily large, interval computations are employed to dismiss parts of the search space that are inconsistent with the constraints. This is the concept of *contractor programming* [9]. A convenient contractor is the so-called *forward-backward contractor*. The idea is to decompose complex constraints into primitive constraints involving only a single arithmetic operation. Due to the decomposition, the solution set can be less accurate than the result of SIVIA but it is computed more efficiently.

For our work, we use interval analysis since

- the true error distribution, which is generally unknown for off-the-shelf sensors, is not required.
- unknown systematic errors (e.g. for the extrinsic calibration between sensors) can be modeled conveniently.
- the results are guaranteed under the assumption that error bounds are not violated.
- no local minima occur and the possibly highly nonlinear functions do not need to be linearized.

Many of these advantages stem from the fact that traditional stochastic methods suffer the convergence and the inconsistency problem [10], meaning that an optimization algorithm

does not always find a global minimum and the different sources of error (e.g. sensor and extrinsic transformation errors) might not follow a Gaussian noise model.

However, to determine the most likely point-valued solution in the interval, a combination with stochastic approaches is required in the future. While the main objective of conventional approaches is to compute the solution, interval analysis can also be understood as a technique to reliably remove inconsistent areas (i.e. areas in which the solution cannot reside). Thus, combining the proposed approach with traditional methods allows to detect failures of these methods. In the context of sensor fusion, this enables the identification of erroneous fused information or, in the context of state estimation, this facilitates the detection of wrong estimates.

IV. Sensor Error Models

We assume bounded-error sensor models which have been first introduced in our most recent papers [2], [11].

A. Camera Model

First, we have to decide which image points we want to augment with depth information. Here, we use "Good features to track" which is a state-of-the-art approach to detect features in the image [12]. Subsequently, we use the Lucas Kanade feature tracker to reidentify these features in consecutive images [13]. In this way, we are able to establish so-called image feature matches, meaning that the matched image features of the consecutive images correspond to the same object. Consequently, the raw measurements we use for sensor fusion are pixel points in the image. Subsequently, we apply the pinhole camera model to find the normalized coordinates (i.e. the z-coordinate is 1) of these features.

However, image features cannot be detected and matched perfectly for different reasons. First, the camera discretizes the actual scene into pixels. Since this discretization results in natural interval bounds (i.e.$[-0.5, 0.5]\,\mathrm{px}$), the interval-based error model is preferable. Second, different factors (e.g. image blur, noise or imprecise intrinsic parameters) distort the feature matching process and/or the computation of the normalized coordinates, but cannot be quantified (i.e. the true error distribution is unknown) since they depend on various circumstances (e.g. the environment) [14]. Assuming to know error bounds is a weaker assumption than assuming to know the full error distribution, and thus the interval-based model is again preferable. For this work, we empirically determine a box $[\Delta_{px}]$ (cf. Fig. 2) that bounds all error influences. In other words, we assume that even in the worst case, the feature matching is not off by more than $[\Delta_{px}]$. Naturally, outliers (i.e. features that are completely mismatched) can still occur and will be considered later.

B. LiDAR Model

Conceptually, a 3D LiDAR measures only a radial distance r and allows to compute the 3D Cartesian point by also supplying the polar (vertical) angle θ and the azimuthal (horizontal) angle φ of the emitted beam. However, this beam is not a perfect beam but diverges, and thus the actually measured point could be anywhere within the resulting beam footprint [15]. Without knowing about the power distribution within the beam, the error distribution of the measured point cannot be inferred. In contrast, the beam divergence (i.e. the bounds) is usually specified by the manufacturer, and thus an interval-based error model is convenient. Besides, the incidence angle of the laser beam and the environmental conditions (e.g. humidity) influence the distance measurement and can lead to a systematic error (bias) [3], [16]. Since the incidence angle and environmental conditions are generally unknown, a stochastic error model is not applicable. In contrast, manufacturers usually specify the maximum error of the distance measurement assuming usual conditions (e.g. $\leq 50\,^{\circ}\mathrm{C}$). Fig. 3 shows our error model.

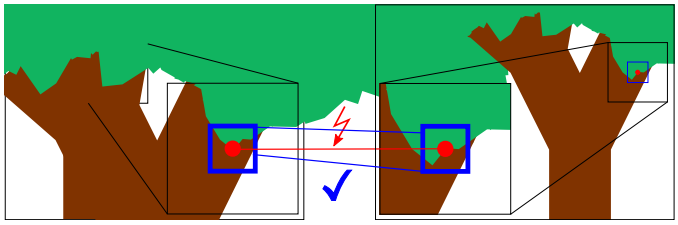


Fig. 2. Exemplary images for which matching image features are sought. The red dots are the anticipated result of traditional image feature matching and do not reside on exactly the same object due to the mentioned error sources. Thus, we use interval boxes (depicted in blue) to model the uncertainties. As can be seen, the two interval boxes overlap for parts of the same object, and therefore establish a correct feature matching.

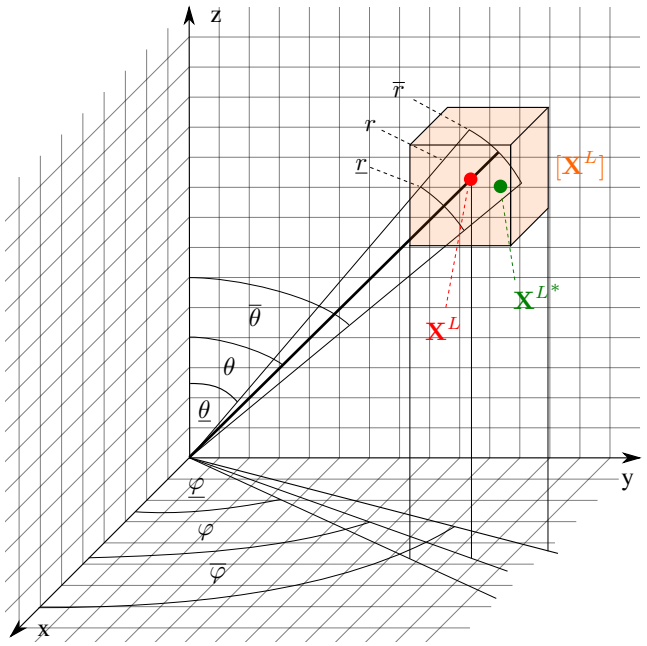


Fig. 3. The spherical coordinate intervals $[r]$, $[\theta]$ and $[\varphi]$ are computed by inflating the raw measurements (r, θ, and φ) by the maximum error specified by the manufacturer (e.g. $[r] \coloneqq r + [\Delta_r]$, where $[\Delta_r]$ encloses the maximum error of the distance measurement). Next, we use the so-called *natural inclusion function*, which is a function that replaces each real operator by its interval counterpart [4], to compute the Cartesian 3D box $[\mathbf{X}^L]$ from these spherical coordinate intervals. Due to some unknown error, the actually measured point $\mathbf{X}^{L*}$ is different from the point $\mathbf{X}^L$ corresponding to the point measurement, but it is enclosed by the computed 3D box.

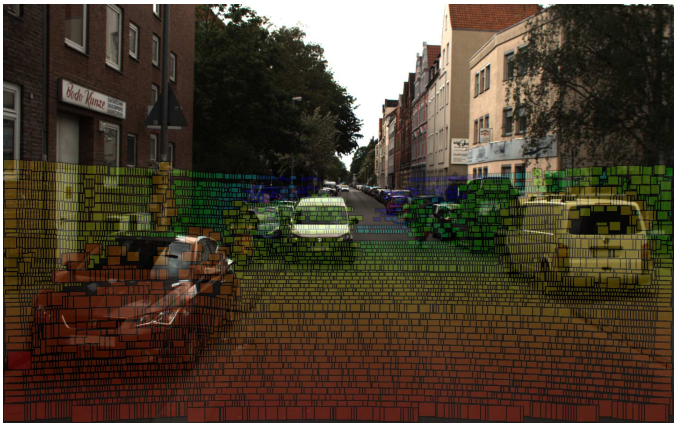


Fig. 4. Interval boxes enclosing the projected LiDAR points. The color of each box corresponds to the midpoint of the depth interval: from red (close) to blue (distant).

V. Visual-LiDAR Sensor Fusion

The purpose of this section is to augment visual features with depth information for subsequent robot localization.

A. Laser scan projection

First, camera and LiDAR information must be brought into a common reference frame, which is the camera coordinate system C for this work. Thus, the extrinsic transformation between the sensor coordinate systems, which consists of the rotation matrix $\mathbf{R}_L^C \in SO(3)$ and the translation vector $\mathbf{t}_L^C$, must be known. Subsequently, we can define the following function to transform a 3D laser scan point $\mathbf{X}_i^L$ from the LiDAR coordinate system L into the camera coordinate system C and project it onto the image plane (i.e. normalize the coordinates such that the z-coordinate is 1):

$$\begin{pmatrix} \tilde{x}_i^C \\ \tilde{y}_i^C \end{pmatrix} = \mathbf{f}_{\text{proj}} \left(\mathbf{X}_i^L, \mathbf{R}_L^C, \mathbf{t}_L^C \right) = \begin{pmatrix} \frac{\mathbf{R}_1 \cdot \mathbf{X}_i^L + t_1}{\mathbf{R}_3 \cdot \mathbf{X}_i^L + t_3} \\ \frac{\mathbf{R}_2 \cdot \mathbf{X}_i^L + t_2}{\mathbf{R}_3 \cdot \mathbf{X}_i^L + t_3} \end{pmatrix}, \tag{3}$$

where $\mathbf{R}_m$ and t_m, $m \in \{1, 2, 3\}$ are the m-th row of $\mathbf{R}_L^C$ and $\mathbf{t}_L^C$, respectively. $(\tilde{x}_i^C \ \tilde{y}_i^C)^\intercal$ are the normalized coordinates (i.e. the z-coordinate is 1 and thus omitted).

The laser scan point and the extrinsic transformation cannot be determined exactly, but are only known up to some accuracy. Thus, we define $[\mathbf{f}]_{\text{proj}}$ as the *natural inclusion function* (cf. [4]) of $\mathbf{f}_{\text{proj}}$. Consequently, we compute the box enclosing the normalized coordinates of the scan point i as

$$[\tilde{\mathbf{X}}_i^C] = \begin{pmatrix} [\tilde{x}_i^C] \\ [\tilde{y}_i^C] \end{pmatrix} = [\mathbf{f}]_{\text{proj}} \left([\mathbf{X}_i^L], [\mathbf{R}_L^C], [\mathbf{t}_L^C] \right), \tag{4}$$

where $[\mathbf{X}_i^L]$ is the 3D box enclosing the scan point i according to the bounded error model derived in Section IV-B. Moreover, $[\mathbf{R}_L^C]$ and $[\mathbf{t}_L^C]$ are the extrinsic parameters, which cannot be determined exactly, but are known up to some accuracy [2].

Fig. 4 shows laser scan boxes that are projected onto the image plane. The depth interval of a projected scan point i is its z-coordinate before the projection, denoted as $[z_i^C]$.

B. Data fusion

After bringing the data from both sensors into a common reference system, we can now augment visual features, which are detected and tracked as detailed in Section IV-A, with depth information. Fig. 1 shows the general idea of our approach for sensor fusion. Since image features and scan points are no longer points but boxes we find all overlapping scan boxes for each image feature box. Consequently, we compute the depth of the image feature as the union over the depth of all intersecting scan boxes. This results in an interval for the depth which allows us to immediately assess its accuracy.

Let j be an image feature and $[\tilde{\mathbf{X}}_j^C] = ([\tilde{x}_j^C] \ [\tilde{y}_j^C])$ its normalized coordinates on the image plane according to the bounded error model introduced in Section IV-A. We determine the set $\mathcal{S}_j$ of all previously projected scan boxes i whose normalized coordinates have a non-empty intersection with the normalized coordinates of feature j:

$$\mathcal{S}_j = \{\, i \in \{1, \ldots, N_l\} \mid [\tilde{\mathbf{X}}_i^C] \cap [\tilde{\mathbf{X}}_j^C] \neq \varnothing \,\}. \tag{5}$$

Subsequently, we can determine the z-coordinate (i.e. the depth) of the image feature j by computing the union over the depth of all overlapping scan boxes:

$$[z_j^C] = \bigcup_{i \in \mathcal{S}_j} [z_i^C]. \tag{6}$$

Next, we use this interval to determine the 3D coordinates:

$$[\mathbf{X}_j^C] = [z_j^C] \begin{pmatrix} [\tilde{x}_j^C] \\ [\tilde{y}_j^C] \\ 1 \end{pmatrix}. \tag{7}$$

Of course, the underlying assumption to fuse data using our proposed method is that the LiDAR measures points close enough to every image feature. Otherwise, it might happen that the true depth of a feature is not enclosed in the computed interval. However, this is the case since the LiDAR produces a dense point cloud and we check that each image feature is surrounded by sufficient 3D points.

The result of this approach are visual features that are augmented by a depth interval which is guaranteed to enclose the true distance of the feature. This allows to directly assess the accuracy of the depth estimates, and thus enables to distinguish between accurate and inaccurate features. Fig. 5 shows exemplary images that illustrate the results of our data fusion approach. Since the projected point cloud does not cover the entire image, we cannot calculate the depth of every feature. Nevertheless, image features without depth information can still be employed for the motion estimation. Thus, in the following we distinguish between features with and without depth information.

VI. Guaranteed Visual-LiDAR Odometry

Next, we aim to incrementally localize a robot using the previously fused information from camera and LiDAR. Here, no global information (e.g. a known map or known pose in a global coordinate system) is available. Thus, we can only compute the relative motion of the robot between the

(a) Colored by depth: from red (close) to blue (distant); pink: no depth.

(b) Colored by depth **accuracy**: from red (accurate) to blue (inaccurate).

Fig. 5. Exemplary results of our approach for data fusion. As expected, the depth uncertainty is larger for features that are close to edges since the feature could lie on either side of the edge (e.g. features on the border of the car).

acquisition of image and laser scan data. Fig. 6 shows the general idea of our approach. To initially constrain the rotation of the robot, we use data from an IMU. This allows to also estimate the robot's motion in disadvantageous feature configurations (e.g. only features on the same plane), but is not necessarily required for our approach. Since the use of the IMU is not the focus of this document, we refer to [11].

Formally, we define the problem of visual-LiDAR odometry as follows. Let f and g be two consecutive image frames. The motion of the robot between these image frames is described by the 6-DOF (degrees of freedom) rigid body transformation consisting of the rotation matrix $\mathbf{R}_g^f$ and the translation vector $\mathbf{t}_g^f$. Here, we omit the superscript indicating the coordinate system and only specify the image frame since all information is already transformed into C:

$$\mathbf{X}_j^f = \mathbf{R}_g^f \mathbf{X}_j^g + \mathbf{t}_g^f, \tag{8}$$

where $\mathbf{X}_j^h$ are the 3D coordinates of the feature j found in image $h \in \{f, g\}$. These 3D coordinates are determined by fusing information from LiDAR and camera as detailed in Section V-B. Consequently, we can formulate the constraint

$$\mathbf{f}_j^{\text{both}}\left(\mathbf{R}_g^f, \mathbf{t}_g^f, \mathbf{X}_j^f, \mathbf{X}_j^g\right) = \mathbf{R}_g^f \mathbf{X}_j^g + \mathbf{t}_g^f - \mathbf{X}_j^f = \mathbf{0}. \tag{9}$$

If we find the depth of a feature only for image g, but not for image f, the problem of estimating the rotation matrix and

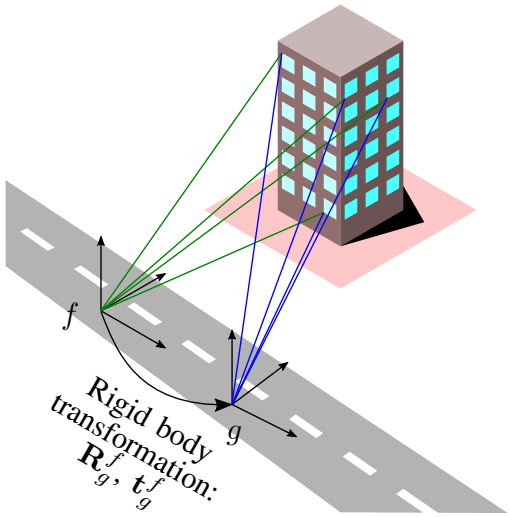


Fig. 6. General idea of our approach to use 3D features to compute the relative motion of a robot.

the translation vector becomes

$$z_j^f \tilde{\mathbf{X}}_j^f = \mathbf{R}_g^f \mathbf{X}_j^g + \mathbf{t}_g^f, \tag{10}$$

where z_j^f is the unknown depth and $\tilde{\mathbf{X}}_j^f$ are the normalized image coordinates. By rewriting (10) and eliminating the unknown z-coordinate z_j^f, we can formulate the constraint:

$$\begin{aligned}\mathbf{f}_j^{\text{one}}\left(\mathbf{R}_g^f, \mathbf{t}_g^f, \tilde{\mathbf{X}}_j^f, \mathbf{X}_j^g\right) &= \begin{pmatrix}(\mathbf{R}_1 - \tilde{x}_j^f \mathbf{R}_3)\mathbf{X}_j^g + t_1 - \tilde{x}_j^f t_3 \\ (\mathbf{R}_2 - \tilde{y}_j^f \mathbf{R}_3)\mathbf{X}_j^g + t_2 - \tilde{y}_j^f t_3\end{pmatrix} \\ &= \mathbf{0},\end{aligned} \tag{11}$$

where $\mathbf{R}_m$ and t_m, $m \in \{1, 2, 3\}$ are the m-th row of $\mathbf{R}_g^f$ and $\mathbf{t}_g^f$, respectively. Similarly, if only the depth of the feature in image g is unknown, a similar constraint can be formulated.

If we fail to find the depth of a feature in both corresponding images, the constraint becomes:

$$\begin{aligned}&\mathbf{f}_j^{\text{no}}\left(\mathbf{R}_g^f, \mathbf{t}_g^f, \tilde{\mathbf{X}}_j^f, \tilde{\mathbf{X}}_j^g\right) = \\ &\begin{pmatrix}-\tilde{y}_j^f T_3 + T_2 & \tilde{x}_j^f T_3 - T_1 & -\tilde{x}_j^f T_2 + \tilde{y}_j^f T_1\end{pmatrix} \mathbf{R}_g^f \tilde{\mathbf{X}}_j^g = 0.\end{aligned} \tag{12}$$

We omit the full derivation due to lack of space.

Subsequently, we stack the resulting constraints (9), (11) and (12) of all image features j into a multi-dimensional constraint $\mathbf{f}$. Since the coordinates of all image features are unknown but bounded, we formulate the problem of computing $\mathbf{R}_g^f$ and $\mathbf{t}_g^f$ as a CSP:

$$\mathcal{C}: \begin{cases}\textbf{Variables: } \mathbf{R}_g^f, \mathbf{t}_g^f \\ \textbf{Constraint: } \mathbf{f}\left(\mathbf{R}_g^f, \mathbf{t}_g^f\right) = \mathbf{0} \\ \textbf{Domains: } [\mathbf{R}_g^f], [\mathbf{t}_g^f]\end{cases}$$

To reduce the number of unknowns to six, the rotation matrix $\mathbf{R}_g^f$ is expressed using three Euler angles $\boldsymbol{\xi}_g^f = (\varphi_g^f \ \theta_g^f \ \psi_g^f)^\mathsf{T}$ defined in the ZYX convention. The initial domains $[\boldsymbol{\xi}_g^f]$ are set using measurements by the IMU. Moreover, the initial domains for the translation are set to $[\mathbf{t}_g^f] = ([-\infty,\infty] \ [-\infty,\infty] \ [0,\infty])^\mathsf{T}$. Thus, we assume no information

other than that the robot is moving forward. Theoretically, the accelerometer of the IMU could be used to compute initial intervals for the translation. However, we refrain from doing so since these intervals would be too inaccurate to help the computation in any significant way.

Afterwards, we build a forward-backward contractor for every constraint of $\mathcal{C}$ using the interval library IBEX [17]. These contractors allow to remove parts of the initially large intervals $[\boldsymbol{\xi}_g^f]$ and $[\mathbf{t}_g^f]$ that are inconsistent with the constraints in $\mathcal{C}$. Finally, only small intervals remain which are guaranteed to contain the true solution.

Keyframes: Our approach is keyframe-based which means that we estimate the transformation from the current image frame g relative to the most recently defined keyframe f until we have to insert a new keyframe [18].

We use a dynamic approach that observes the uncertainty of odometry estimates (i.e. the radius of $[\boldsymbol{\xi}_g^f]$ and $[\mathbf{t}_g^f]$) and inserts a new keyframe once the pose uncertainty exceeds a predefined threshold. An increasing uncertainty is the direct consequence of insufficient constraints, which prevent our contractors from contracting the pose intervals, and thus a direct indicator of insufficient visual features. This allows us to insert keyframes only when it is absolutely necessary.

Outlier Treatment: Although our approach to fuse data from camera and LiDAR is guaranteed to compute a proper enclosure for the depth of a visual feature, outliers can occur during visual feature matching or due to moving objects. Thus, we have to account for them using a relaxed intersection [4]. This means that not all constraints of the CSP $\mathcal{C}$ have to be met, but a predefined number can be inconsistent. To avoid groups of outliers that are mismatched for the same reason (e.g. on repetitive structures like fences) and which would make it difficult bound the number of outliers, we divide the image into small sub-regions, in each of which we only select a predefined number of features.

VII. Experiments and Discussion

Our data set, which we use for the following evaluation, is freely available [19]. It has many tall buildings along the route that prevent GNSS information but offer many visual features. We employed the multi-sensor platform that can be seen in Fig. 7. Among other sensors, this platform houses a LORD MicroStrain 3DM-GQ4-45 IMU, a Velodyne HDL-64E LiDAR and a FLIR Grasshopper3 color camera. To measure accurate ground truth information, the vehicle is additionally equipped with a Riegl VMX-250 Mobile Mapping System that incorporates a highly accurate GNSS/IMU system. The camera is set to capture images whenever the LiDAR faces forward, resulting in a frame rate of 10 frames per second. More details can be found on our website [19].

The uncertainties of the LiDAR are specified by the manufacturer as $[\Delta_r] = [-6, 6]\,\mathrm{cm}$ and $[\Delta_\theta] = [\Delta_\varphi] = [-1.5, 1.5]\,\mathrm{mrad}$. The empirically determined error bounds for the image feature matches are composed of a fixed uncertainty $[\Delta_{px_f}] = [-2, 2]\,\mathrm{px}$ and an uncertainty $[\Delta_{px_s}] = [-0.1, 0.1]\,\mathrm{px}$ that scales with the distance the vehicle moved between image acquisitions. In addition, we choose a maximum of $5\,\%$ outliers for the relaxed intersection explained

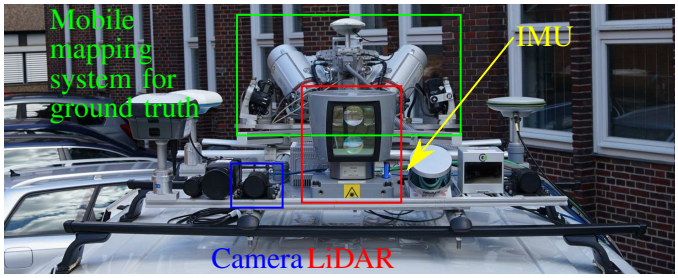


Fig. 7. Our multi-sensor platform. Image credit: Sören Vogel.

in Section V. We determined this percentage empirically (by evaluating the number of matches that are inconsistent with the ground truth trajectory of different test drives) and selected it rather conservatively to keep the guarantees intact. Besides, we insert a new keyframe once the area of the 3D position box projected to the ground plane exceeds $5\,\mathrm{m}^2$.

Since our approach only serves the purpose of computing the robot's pose relative to the last keyframe, we only evaluate this relative, but no global localization results. This means that the uncertainty that has accumulated when inserting a new keyframe is not further propagated but reset. If we would try to localize the robot relative to the start of the experiment, the uncertainty would grow infinitely due to drift. This is because we have no global information to reduce the localization error that has accumulated once we have to set a new keyframe, but only use information from two distinct points in time. Thus, global contractors should be developed in the future to prevent the uncertainty from increasing rapidly. These global contractors can be found, for example, by extending our approach by bundle adjustment, extending it to SLAM, or by using GNSS information.

We show the advantages that combining our approach with the optimization method proposed by Zhang et al. [6] can bring. For this, we only implement their frame to frame motion estimation since our approach does not perform bundle adjustment. Besides, we provide both our and their approach with the same image features and point clouds.

We first show quantitative results for both our and the KITTI data set [20] in Table I. From the KITTI data set, we use different sequences, which are indicated in the first row of the table as four-digit numbers (all from 2011/09/26). Since the same sensors are used but only few information about the errors are provided by the authors, we use the same error bounds as for our data set. Note that we cannot provide the standard evaluation metrics such as the mean error or standard deviation. Our interval-based approach computes the 6-DOF pose boxes $[\boldsymbol{\xi}_g^f]$ and $[\mathbf{t}_g^f]$ that are guaranteed to contain the true solution without being able to judge which point-valued pose is the most likely solution (cf. Section III).

As can be seen, we are able to reliably enclose the robot's true pose for all 2644 image frames of our own data set, and are thus able to provide the desired guarantees. Although the numbers do not indicate that these guarantees can also be provided for the KITTI data set, we are convinced that the few faults (25 out of 1155 frames) are either due to the missing information about the sensor and extrinsic calibration errors

TABLE I
QUANTITATIVE RESULTS OF OUR APPROACH.

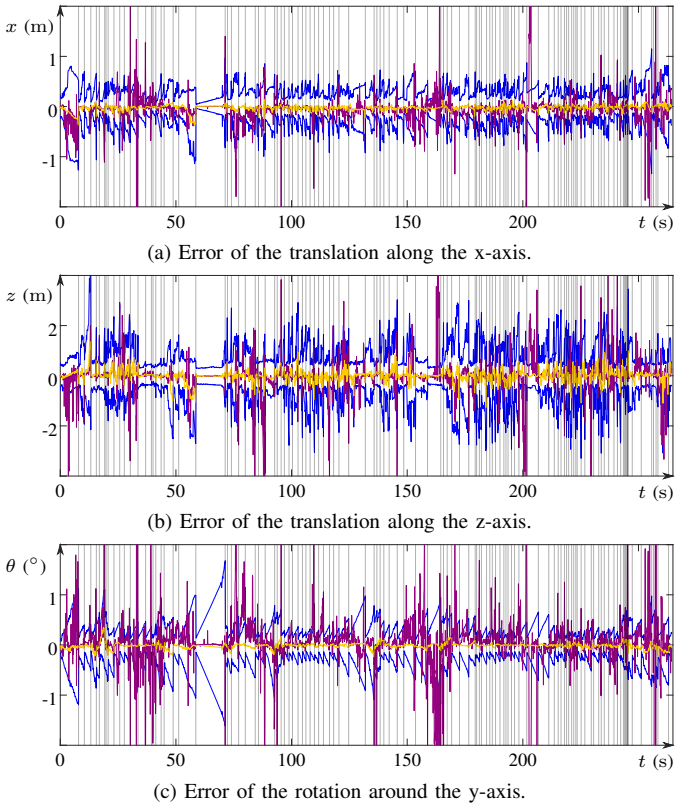
Metric	[19]	0009 [20]	0023 [20]	0106 [20]
Percentage of image frames for which the true pose is enclosed in the computed 6-DOF pose boxes	100 %	97.8 %*	97.7 %*	98.3 %*
Average volume of the 3D position boxes	0.77 m^3	1.52 m^3	0.91 m^3	0.94 m^3
Average area of the 3D boxes on the ground	1.30 m^2	1.81 m^2	1.39 m^2	1.32 m^2
Average radius of the interval enclosing the robot's orientation on the ground plane	0.39°	0.21°	0.19°	0.22°
Average number of image features with depth	105	201	173	170
Average distance after which we insert a new keyframe due to the uncertainty becoming too large	11.35 m	9.57 m	11.91 m	11.48 m
Percentage of image frames for which the result of [6] is not enclosed in the 6-DOF pose boxes	29.9 %	16.1 %	18.2 %	12.2 %

or due to incorrect ground truth poses. This is also evident by the fact that many faults occur in direct succession, which can be attributed to brief GNSS outages.

Moreover, the accuracy of our approach is reasonable, considering that we take the maximum error into account and allow up to 5 % outliers. Besides, we are able to detect an inconsistency of the state-of-the-art approach for up to three out of ten image frames, showing that our approach is more robust. While the approach of Zhang et al. is real-time capable and takes approximately 40 ms per image frame, our approach has not yet been optimized and needs on average 0.9 s per image frame. This is not due to the design of our approach (we use efficient contractors [9]) but due to the implementation, which should be optimized in the future.

Fig. 8 shows results with respect to ground truth information for our own data set [19]. Since the car moves mainly in two dimensions, we only show results for the translation in x- and z-direction and the rotation around the y-axis. It is evident that our error bounds never violate the true result (i.e. 0 is always enclosed in those intervals). Furthermore, the midpoint of our intervals approximates the true result reasonably well, and thus it can be used if a point-valued result is needed. Next, we want to point out that the uncertainty of the localization results varies drastically. Consequently, keyframes are inserted more frequently for some parts of the trajectory. This is due to the fact that different areas of the trajectory provide 3D features of different numbers and different uncertainties. A similar behavior is difficult to achieve using established stochastic approaches if the uncertainties of the features are neglected, and thus the accuracy of the pose estimates cannot be assessed.

In addition, Fig. 8 illustrates the inconsistencies between the results of the stochastic and our interval-based approach. Whenever the result of the stochastic approach lies outside the lower and upper interval bounds (i.e. the violet line crosses a blue line), the stochastic solution cannot be correct and therefore an inconsistency has occurred. We believe that these inconsistencies occur since the stochastic approach disregards any uncertainty the augmented features might have. Thus, better features cannot be assigned a higher weight, and therefore many bad features overrule the few good features. In contrast, our interval-based approach considers the uncertainties during sensor fusion. Features for which depth cannot be determined accurately (e.g. on borders) are assigned large depth intervals and vice versa. Consequently, uncertain features contribute significantly weaker constraints to the pose computation than accurate features.



(a) Error of the translation along the x-axis.

(b) Error of the translation along the z-axis.

(c) Error of the rotation around the y-axis.

Fig. 8. Relative error (ground truth is 0) over time for [19]. The x-axis is pointing to the right, the y-axis is pointing downwards and the z-axis is pointing in driving direction. A vertical gray line indicates the insertion of a new keyframe. Thus, pose estimates are always relative to the last vertical line. The lower and upper bounds of our method are depicted in blue, the corresponding midpoints are colored yellow and the result of the stochastic method is colored violet.

Fig. 9 shows augmented feature images for both our and the stochastic approach during one exemplary detected inconsistency. First, it can be seen that our approach uses some features without depth, for which the stochastic approach finds depth. This is because our approach is able to dynamically check if the image feature interval is completely covered by scan point intervals and does not assign depth information if this

*The computed pose boxes do not enclose the ground truth for some frames for which we deem the ground truth information unreliable. We inspected all such cases manually and found that the computed boxes are reasonable whereas the ground truth pose seems off (e.g. due to GNSS outages).

(a) Stochastic features colored by depth: red (close) to blue (distant); pink: no depth.

(b) Interval features colored by middle depth: red (close) to blue (distant); pink: no depth.

(c) Interval features colored by depth uncertainty: red (accurate) to blue (inaccurate).

Fig. 9. Depth augmented image features for the detected inconsistency around $t = 200\,\mathrm{s}$.

is not the case. In contrast, the conventional approach only requires three scan points to be close enough to the image feature. Except for these few features, it can be seen that the depth estimates by the stochastic approach (cf. Fig. 9a) look similar to the mid points of our depth intervals (cf. Fig. 9b). However, Fig. 9c indicates that only a few features are accurate and should be trusted.

Although stochastic approaches successfully apply various methods to cope with erroneous pose estimates (e.g. Zhang et al. employ bundle adjustment to smooth the computed trajectory), we still believe that detecting these faults in a guaranteed way allows to be more robust.

VIII. Conclusions and Future Work

We present an interval-based approach to fuse data from camera and LiDAR in a guaranteed way. Here, the goal is to create visual features that are augmented by distances measured by the LiDAR. In contrast to conventional approaches, we propagate sensor and transformation uncertainties from input sources to the final 3D features. This allows us to perform visual-LiDAR odometry which takes the uncertainty of features into account and propagates it to the final pose interval. Since we use a bounded-error method, our fused 3D features and pose estimates are guaranteed under the given assumptions. The evaluation shows that we are indeed able to reliably enclose the true localization results. Besides, we are able to detect inconsistencies of an established approach for depth-aided visual odometry that does not take the uncertainty during sensor fusion into account.

As future work, we aim to find reliable point-valued results by restricting the stochastic approach to a solution inside our guaranteed intervals. Besides, we plan to extend our motion estimation by incorporating global constraints such as GNSS information or by simultaneously building a map (e.g. based on LiDAR points similar to [21]) that allows us to find additional constraints (i.e. extension to SLAM).

References

[1] H. Strasdat, J. M. M. Montiel, and A. Davison, "Scale Drift-Aware Large Scale Monocular SLAM," in *Proceedings of Robotics: Science and Systems*, Zaragoza, Spain, Jun. 2010.

[2] R. Voges and B. Wagner, "Set-Membership Extrinsic Calibration of a 3D LiDAR and a Camera," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Virtual, Oct. 2020.

[3] Y. Park, S. Yun, C. Won, K. Cho, K. Um, and S. Sim, "Calibration between Color Camera and 3D LIDAR Instruments with a Polygonal Planar Board," *Sensors*, vol. 14, no. 3, pp. 5333–5353, Mar. 2014.

[4] L. Jaulin, M. Kieffer, O. Didrit, and É. Walter, *Applied Interval Analysis*. Springer London, 2001.

[5] X. Zuo, P. Geneva, W. Lee, Y. Liu, and G. Huang, "LIC-Fusion: LiDAR-Inertial-Camera Odometry," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Macau, China, Nov. 2019.

[6] J. Zhang, M. Kaess, and S. Singh, "A real-time method for depth enhanced visual odometry," *Auton Robot*, Dec. 2015.

[7] J. Graeter, A. Wilczynski, and M. Lauer, "LIMO: Lidar-Monocular Visual Odometry," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Madrid, Spain, Oct. 2018.

[8] F. Andert, N. Ammann, S. Krause, S. Lorenz, D. Bratanov, and L. Mejias, "Optical-Aided Aircraft Navigation using Decoupled Visual SLAM with Range Sensor Augmentation," *Journal of Intelligent & Robotic Systems*, vol. 88, no. 2-4, pp. 547–565, Jan. 2017.

[9] G. Chabert and L. Jaulin, "Contractor programming," *Artificial Intelligence*, vol. 173, no. 11, pp. 1079–1100, Jul. 2009.

[10] S. Huang and G. Dissanayake, "A critique of current developments in simultaneous localization and mapping," *International Journal of Advanced Robotic Systems*, vol. 13, no. 5, pp. 1–13, Sep. 2016.

[11] R. Voges and B. Wagner, "Timestamp Offset Calibration for an IMU-Camera System Under Interval Uncertainty," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Madrid, Spain, Oct. 2018.

[12] J. Shi and Tomasi, "Good features to track," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Seattle, WA, USA, Jun. 1994.

[13] J.-Y. Bouguet, "Pyramidal implementation of the Lucas Kanade feature tracker," Microprocessor Research Labs, Intel Corp., Tech. Rep., 2000.

[14] B. T. Koik and H. Ibrahim, "A Literature Survey on Blur Detection Algorithms for Digital Imaging," in *1st International Conference on Artificial Intelligence, Modelling and Simulation*, Kota Kinabalu, Malaysia, Dec. 2013.

[15] Q. Pentek, T. Allouis, O. Strauss, and C. Fiorio, "Developing And Validating a Predictive Model of Measurement Uncertainty For Multi-beam LiDARs: Application to the Velodyne VLP-16," in *International Conference on Image Processing Theory, Tools and Applications (IPTA)*, Xi'an, China, Nov. 2018.

[16] Q. Wang, N. Zissler, and R. Holden, "Evaluate error sources and uncertainty in large scale measurement systems," *Robotics and Computer-Integrated Manufacturing*, vol. 29, no. 1, pp. 1–11, Feb. 2013.

[17] G. Chabert, "IBEX, a C++ library for constraint processing over real numbers," http://www.ibex-lib.org, 2017.

[18] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale, "Keyframe-Based Visual-Inertial Odometry Using Nonlinear Optimization," *The International Journal of Robotics Research*, vol. 34, no. 3, pp. 314–334, Dec. 2014.

[19] R. Voges, "Dataset: i.c.sens Visual-Inertial-LiDAR Dataset," https://doi.org/10.25835/0026408, 2020.

[20] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, Aug. 2013.

[21] B. Desrochers, S. Lacroix, and L. Jaulin, "Set-membership approach to the kidnapped robot problem," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Hamburg, Germany, 2015.