Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Autonomous Systems Lab
Prof. Roland Siegwart

Master Thesis

Odometry and mapping inside pipes

Visual, inertial and kinematic sensor fusion

Spring Term 2016

Supervised by:
Roland Siegwart
Howie Choset

Author:
Elena Morara

Declaration of Originality

I hereby declare that the written work I have submitted entitled

Your Project Title

is original work which I alone have authored and which is written in my own words.[1]

Author(s)

Elena	Morara

Student supervisor(s)

Roland	Siegwart
Howie	Choset

Supervising lecturer

Roland	Siegwart

With the signature I declare that I have been informed regarding normal academic citation rules and that I have read and understood the information on 'Citation etiquette' (`https://www.ethz.ch/content/dam/ethz/main/education/rechtliches-abschluesse/leistungskontrollen/plagiarism-citationetiquette.pdf`). The citation conventions usual to the discipline in question here have been respected.

The above written work may be tested electronically for plagiarism.

Bologna, 2 September 2016
Place and date

Elena Morara
Signature

[1] Co-authored work: The signatures of all authors are required. Each signature attests to the originality of the entire piece of written work in its final form.

Contents

Abstract

This work aims at estimating the motion of a snake-robot while it locomotes through a pipe network and at retrieving a high-level map of the traversed piping. To this end, I designed and implemented a motion estimation algorithm which relies on the readings from a frontal monocular camera, as well as IMUs and encoders along the robot body. Specifically, I devised two novel estimators which exploit the characteristics of the problem statement: the circular features which are visible from the interior of the pipe when two pipe segments are connected together and the shape of the robot body which complies with the piping geometry. I formulated a sensor fusion method which merges my two original estimation techniques together with feature-based visual odometry and inertia-based odometry. Empirical results show that the proposed method outperforms standard visual-inertial odometry approaches. The modularity of my framework makes it flexible so that different robot architectures can benefit from its components.

Symbols

Symbols

${}^{i}T_j$	homogeneous transformation from i to j reference frames
${}^{i}R_j$	direction cosine matrix describing rotation from i to j reference frames
${}^{i}t_j$	translation from i to j reference frames
${}^{i}q_j$	unitary quaternion equivalent to ${}^{i}R_j$
$\otimes$	quaternion (Hamiltonian) product
^{o}	degrees
$ln(\cdot)$	natural logarithm function
$det(\cdot)$	determinant of a matrix
$tr(\cdot)$	trace of a matrix
$(\cdot)^T$	transposed of a matrix
$rot(\alpha, v)$	rotation of the angle α around the vector v
$(\cdot)'$	value at the previous iteration of the algorithm
$\varDelta(\cdot)$	difference between two subsequent iterations of the algorithm
A(a:b,c:d)	submatrix consisting of intersection a-th to b-th rows and c-th to d-th columns of matrix A
$\left.\frac{\partial f(\cdot)}{\partial \cdot}\right\|_{a}$	partial derivative of function f computed at point a
ξ	filter state vector
ζ	filter measurement vector

Indices

x	x axis
y	y axis
z	z axis
cam	camera reference frame
$world$	inertial world reference frame
CF	Complementary Filter reference frame
$pipe$	current pipe reference frame
c	circle
cc	circle center
pp	principle point of the camera

Acronyms and Abbreviations

ETH	Eidgenössische Technische Hochschule
CMU	Carnegie Mellon University
EKF	Extended Kalman Filter
IMU	Inertial Measurement Unit
UKF	Unscented Kalman Filter
CF	Complementary Filter
PDAF	Probabilistic Data Association Filter
DoF	Degree(s) of Freedom
FPS	Frames(s) Per Second
RMSE	Root Mean Square Error
nD	n-dimensional
RANSAC	RANdom SAmple Consensus
SLAM	Simultaneous Localization and Mapping

Chapter 1

Introduction

Motivation

The inspection of the internal conditions of pipes is a task which is often times difficult and dangerous for human operators because of the narrow spaces and inaccessible locations.

Highly-articulated mechanisms like snake robots are a preferred architecture for the automatic inspection of pipes since their flexible bodies allows them to adapt their shape to different pipe network characteristics, such as the diameter of the pipes, the angle of the bends and the vertical/horizontal pipe orientations.

The automatic inspection usually relies on a robot carrying a video camera whose footage is observed by an operator who detects interesting features, such as faults or clogs. Hence, it is crucial to be able to compute an estimate of the robot position over time, so that the detected features can be localized in the pipe network, allowing for an efficient intervention.

Traditional approaches

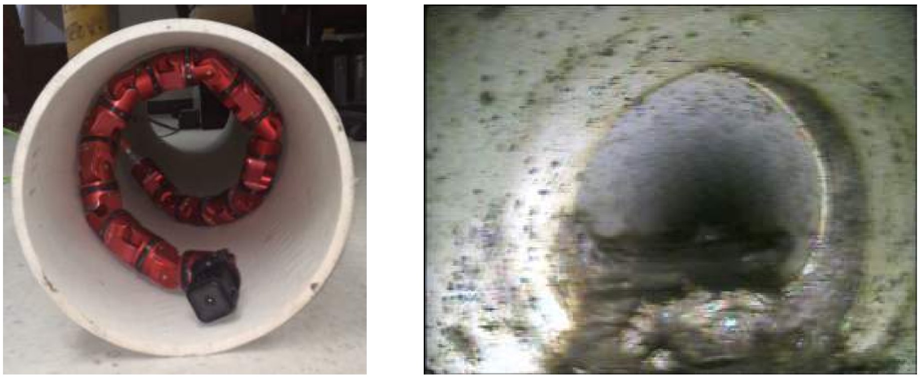
The estimate of the robot motion through dead-reckoning from conventional contact-based motion models shows poor performance because of the slippage due to the smooth and/or wet pipe walls.

As far as traditional visual-inertial odometry approaches are concerned, they are impaired by the very structure of the addressed problem. First of all, the locomoting gait of snake robots poses harsh challenges to inertial position estimation, because the motion of the robot internal degrees of freedom results into a complex motion of both the camera and the robot body parts. Furthermore, the visual part of the algorithm is additionally impaired by the lack of high-quality visual corners and by the changes in illumination due to the robot self-carrying light source.

Proposed algorithm

In order to compensate the flaws of these traditional approaches, I devised two novel estimators to exploit unique characteristics of a pipe network. One of them estimates the robot motion from the virtual motion of circular joints which can be seen at the intersections of two pipes. The other estimator exploits the robot body shape to infer information about the geometry of the environment.

The proposed framework merges the motion estimates from the visual, the inertial and the two novel estimators through an Extended Kalman Filter. The output of the solution is the 6-DoF pose estimate of the robot over time as well as the map of the traversed pipe network.



(a)

(b)

Figure 1.1: Snake robot of Carnegie Mellon University BioRobotics Lab (a) View of the pipe cross section while the robot is traversing it (b) Sample frame of the video footage from the robot frontal camera during an inspection of a drain

I will show some experimental results of my framework on sample setups. While the current solution is customized for a snake robot, the proposed framework can be straightforwardly used with different robot architectures, as long as their body shape adapts to the enclosed space of the pipe they are navigating.

Chapter 2

Literature review

Snake robots in pipes

Previous work on snake robots inside pipes has been mainly focused on the control of the robot internal degrees of freedom to propel itself inside the pipe, see eg. [1] and [2]. A notable exception is the work from Enner, Rollinson and Choset [3], which addresses the problem of estimating the motion of a snake robot in a straight pipe relying on a no-slip assumption. This work distances itself from theirs in that it relaxes the non-slipping assumption and considers the motion trough pipe bends.

Visual-inertial odometry

The merging of visual and inertial information is an established solution to the problem of estimating the motion of a mobile robot. Its popularity comes from the availability of low cost and high performance visual and inertial sensor and the complementarity of the information they can retrieve. This characteristic is the basis for their sensor fusion, whose aim is to leverage the individual benefits of one sensor modality to compensate for the deficiencies of the other. Corke, Lobo and Dias compiled a comprehensive evaluation of the topic in [4]. The inertial information usually comes from an Inertial Measurement Unit (IMU), while the visual sensing is performed by either an on-board monocular or stereo camera. The latter allows triangulation of visual features for every stereo pair, while the former needs to retrieve the 3D structure of the environment from subsequent camera frames, introducing the uncertainty of the camera motion estimate. Either the registration of three-dimensional points or the use of the trifocal tensor is necessary for the computation of the relative scale of subsequent translations of the camera such that the only a scale factor is separating the visual linear motion estimate and the metric one.

As far as the structure-from-motion estimate is concerned, a common broad distinction is among feature-based and appearance-based methods (or dense-motion or optical-flow methods according to [5]). Furthermore, the visual information can be used for either simple visual odometry or simultaneous-localization-and-mapping. As far as visual odometry is concerned, the work of Nister [6] is considered the breakthrough in the field and the basis for many successful systems, while a popular choice for monocular SLAM is the keyframe-based solution of Klein and Murray [7]. For a more in-depth review of the topic the authors refer to the tutorial on Visual Odometry by Scaramuzza and Fraundorfer [8] and [9]. In this work, a visual odometry solution is chosen for its simplicity of implementation.

Sensor fusion

The methods for merging the visual and inertial information can be broadly classified as using tight or loose coupling. In loosely coupled approaches the single sensor modalities independently produce motion estimates and their outcomes are merged in a second step. On the contrary, in tightly coupled approaches the sensor data are fused at a prior stage, allowing for the exploitation of cross correlation between information but also increasing the complexity of the system. This works implements loosely coupled sensor fusion.

Most common sensor fusion methods either use filtering or optimization techniques. The batch optimization methods aim at minimizing the re-projection errors of visual features and the inertial temporal error at the same stage, while the filtering methods are recursive state-propagation-and-update techniques. See the review on the field by Gui et al. [10] for in-depth analysis.

As far as the filter fusion of the information, EKF and UKF are very popular choices. The state to be observed usually includes the position, the velocity and the orientation of the moving object, as well as the scaling factor between the visual and inertial estimates of the translation. System parameters can also be incorporated into the state, as it often happens with the biases affecting the gyroscope and acceleration measurements and the relative pose between camera an IMU (e.g. [11] and [12]). More rarely the state comprises the linear acceleration and the angular velocity (e.g. [13]) of the robot. In tightly coupled approaches, the state is augmented with either the 2D visual feature correspondences or their 3D triangulated anti-projections. As far as the motion model is concerned, the most popular solution consists in using an IMU propagation model, which entails that the process model is replaced by integrated inertial observations. This is the case, among the others, of [14], [15], [16] [11], [17], [18], [19], [20], [21] and [22]. The underlying intuition is that the state prediction step should be based on the readings of proprioceptive sensors, while the exteroceptive sensors are used in the measurement update. According to Lupton [23], historically the method is born from aerospace inertial navigation.

Other works use a constant velocity, acceleration or jerk motion model and both the IMU and visual observation are considered as measurements, e.g. Nutzi et al [24] and Strelow and Singh [25].

A unique solution is the one from Konolige in [5], who uses visual odometry for a data-driven motion model and the inertial information for the measurement update. Another peculiarity of Konolige is the fact that he pre-processes the IMU readings into a pose estimate before feeding it into the sensor fusion filter.

This work is close to the one of Konolige in two aspects: first is that I use visual odometry estimates in the prediction step of the filter, and secondly I also implemented a pre-EKF Complementary Filter.

As far as the problem of retrieving the scale of the visual odometry translation is concerned, the most popular solution is to track its evolution over time by including it in the filter state. It is racked by either double integrating the accelerometer readings (e.g. as in [11]), or by integrating the accelerometers ones and differentiating the visual odometry position (e.g. as in [26]).

The EKF is often implemented as an error-driven filter. Indirect filters facilitate the dealing of quaternions in the filter state, as shown in [27].

Chapter 3

Problem statement

This chapter introduces to the problem considered in this work, whose setup consists of a snake robot and a pipe network. A sample setup is shown in Figure 3.1. Section 3.1 introduces the assumptions on the pipe network, while snake robot characteristics and motion are discussed in Section 3.2. Section 3.3 describes the reference frames which are used in the proposed algorithm as well as the general notation.

3.1 Pipe network

The proposed solution is aimed at pipe networks made by straight pipes connected together. The connection among pipes can be due to sealing or connecting elements. Connecting elements can be either straight junctions, bends or fork junctions. Figure 3.2 shows sample connecting elements for PVC pipes. From now on in this work, forks are referred to as either bends or straight junctions according to the path that the robot follows. The circle which is visible at the location where two pipes or a pipe and a connecting element are joined together is referred to as 'pipe joint'.

This work requires the length of bends to be significantly shorter than the robot length. This is required in order to retrieve information about the geometry of the environment from the robot shape.

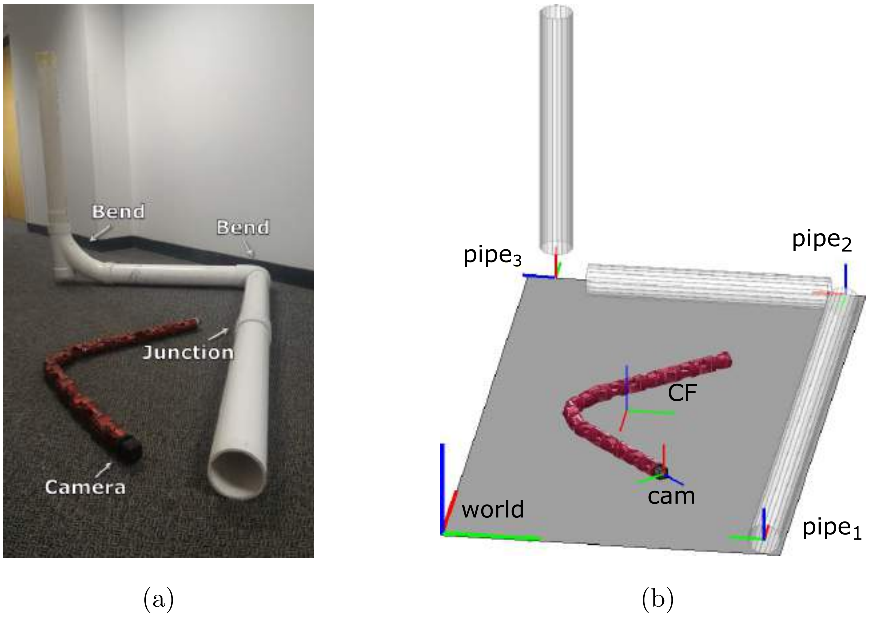
I also assume that the internal walls of the pipes are not completely visually homogeneous, which is a reasonable assumption for many pipe inspection real-world applications.

I will refer to pipe segment as every portion of the pipe network which is straight, irrespectively of how many pipes connected though straight junction it is made of.

Finally, the length of the pipe segments is assumed to be such that the robot body does not go through more than two segments at the same time.

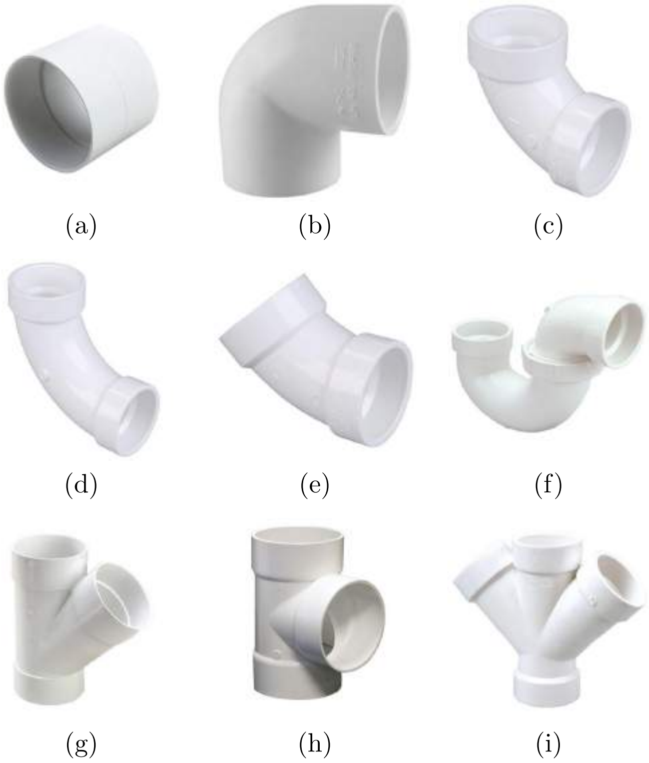
3.2 Snake robot

In this work, I have been using SEA-snake robots from the BioRobotics Insitute of CMU. They are a type of hyper-redundant robots consisting of a serial chain of identical modules, shown in Figure 3.3(a). Each module has one degree of freedom and it is equipped with an encoder, a torque sensor and an IMU with a 3-axis accelerometer and a 3-axis gyro. The rotation joint is actuated by a Series Elastic Actuator. The snake is mounted such that subsequent joint axes are perpendicular to each other, allowing for a three-dimensional motion in the space, as shown in



(a) (b)

Figure 3.1: Sample setup consisting of a snake robot and a three-dimensional pipe network (a) Picture from a real setup (b) Reproduction of the same setup with the reference frames highlighted

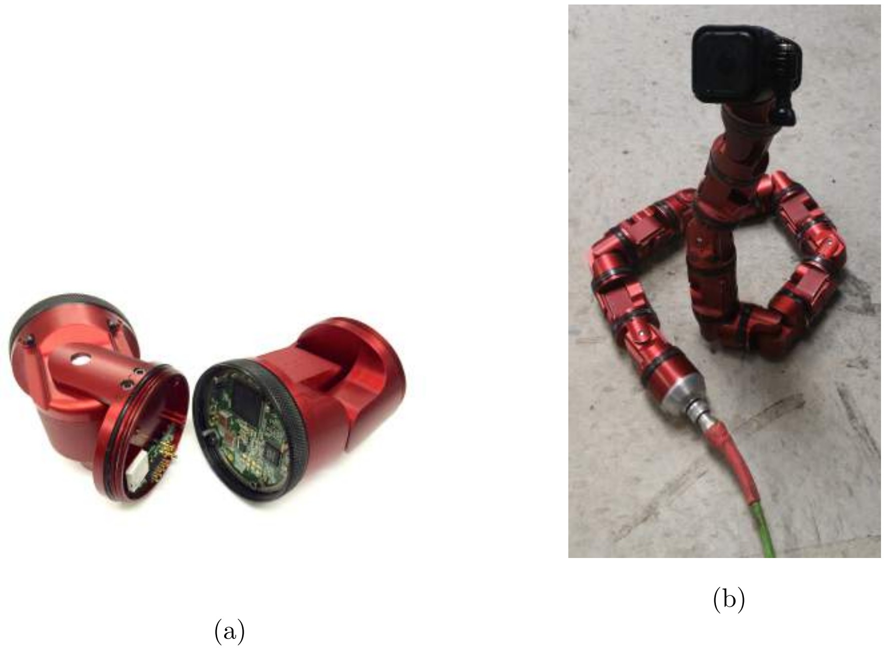


(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

Figure 3.2: Sample PVC connections used in the experimental results of this work. Connection (a) is a straight junction. Connections (b), (c), (d), (e) and (f) are bends. Connections (g), (h) and (i) are forks



(a)

(b)

Figure 3.3: Snake robot used in this work (a) Two of the identical modules which constitute the robot body (figure from [28]) (b) Snake robot equipped with a frontal camera

Figure 3.3(b). For further details on the snake robot architecture, the author refers to [28].
The snake robot is also equipped with a frontal monocular camera, whose intrinsic camera parameters are known.
The robot carries a light source secured next to the camera.

Robot motion

Snake robots locomote through pipe networks by taking the shape of a helix whose radius is slightly larger than the traversed pipe one and exploiting the friction against the pipe walls. The robot moves thanks to a scripted cyclical motion of its internal degrees of freedom called 'rolling gait'. It consists in sinusoidal motion of the joints such that the ones actuating the dorsal direction are at their maximum displacement when the ones actuating the lateral direction are zero and viceversa. See the work from Tesch et al. [29] for an in-depth discussion on this and others gaits for the snake robot. Furthermore, it is straightforward to implement a rolling gait depending on the desired helix radius and pitch thanks to recent work from Zhen, Gong and Choset [1], who generated control strategies directly based on the shape of a backbone curve in the workspace. Figure 3.4 shows the snake robot crawling up a vertical pipe.
I modified the nominal rolling gait so that the frontal camera of the robot would point towards the motion direction and would approximately lie on the pipe main axis. This leads to a significant improvement in the performance of the visual odometry part of my estimation algorithm, for two reasons. First, the Feature-based Visual Odometry benefits from the bigger amount of shared visual features between subsequent frames. Secondly, the Hough detector in the Visual Joint Tracker is more likely to detect an accurate pipe-joint circle, since it is in the camera view most of the time and its appearance is not deformed by the skewed perspective (see

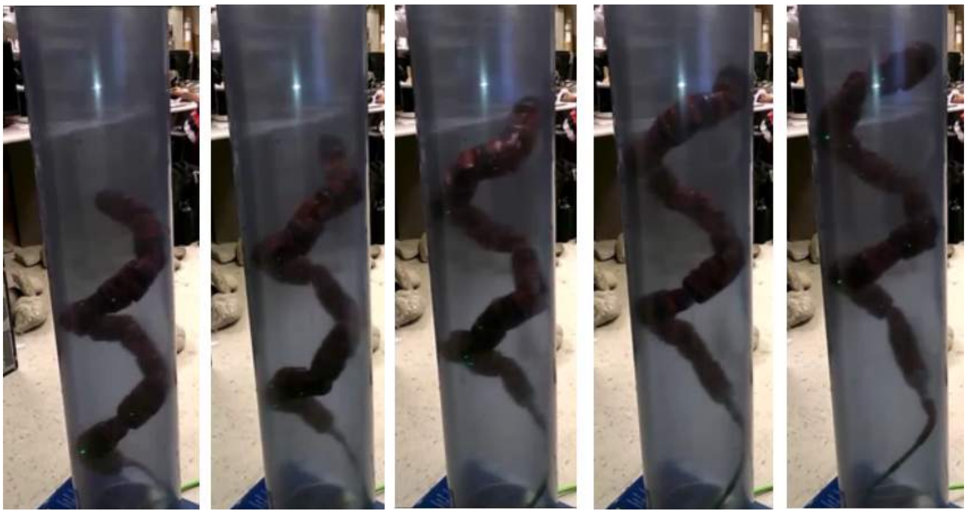


Figure 3.4: Snake robot crawling up a pipe. It climbs 0.5 meters in 8 seconds

Section 4.4 for details). Figure 3.5 shows a comparison between the video from the frontal camera when the robot is locomoting with a nominal rolling gait and a rolling gait with rectified head.

The last three modules of the snake are devoted to the rectification of the head. Their motion is computed by solving an optimization problem. I performed a dense search in the joint space and minimized the projection of the head distance from the pipe axis as well as the mis-alignment between the pipe axis and the head forward direction at discrete instants of the nominal rolling gait.

When the robot meets a bend, the passive compliance of the actuators allows the body shape to adapt to the bend while locomoting though it, as shown in Figure 3.6.

3.3 Reference frames and notation

Figure 3.1 shows the reference frames used in this work.

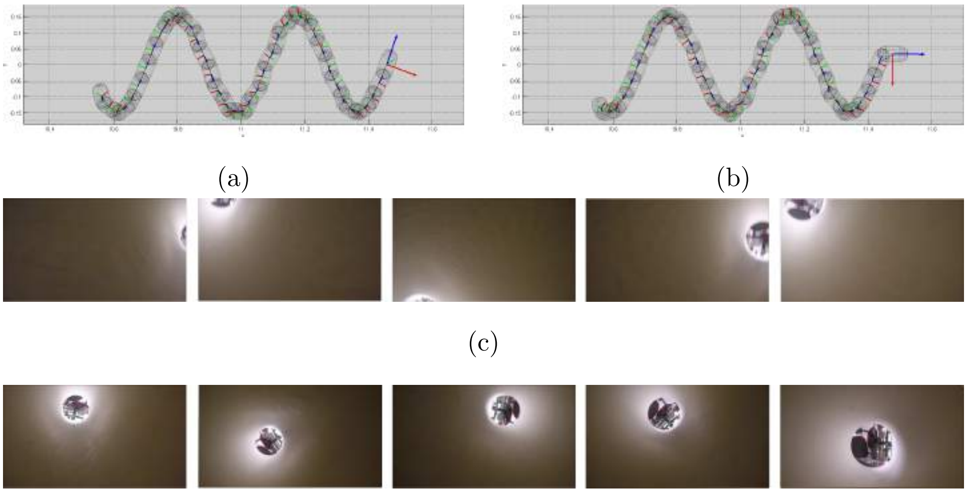
The reference frame *cam* is located at the camera center, with its z-axis pointing forward and its x-axis pointing upwards in the camera view. Since the camera is rigidly attached to the robot, it follows that estimating the pose of *cam* is equivalent to estimating the pose of the robot.

The inertial reference frame *world* is the one with respect of which the motion of the robot has to be estimated; it has its z-axis pointing in the opposite direction of the gravity vector.

I consider also the inertial frames $pipe_i$, $i = 1, \dots, N$,with N the number of pipe segments in the traversed pipe network, which are oriented with the x axis along the pipe direction. For each point in time, I refer to the reference frame of the pipe segment in which the camera is located as ’current pipe’, simply denoted as *pipe*.

The position in metric units of the center of a pipe joint is referred to as $joint$.

Lastly, let us consider a non-conventional frame CF, whose origin is at the instantaneous kinematic center of mass of the robot, and whose orientation is dictated by the robot orientation estimate of the Complementary Filter. In an ideal case, the orientation of CF is temporally invariant with respect to the inertial frames, but in practice it is affected by the unavoidable orientation estimate drift due to the gyro readings.



(a)

(b)

(c)

(d)

Figure 3.5: Role of the rectification of the robot head during its locomotion (a) Side view of the robot inside a pipe during the nominal rolling gait. The head is on the right of the figure, with a bigger reference frame (b) Pose of the robot during the rolling gait with rectification of the head. The head is now pointing towards the locomotion direction. (c) Sample frames of the video from the frontal camera recorded while the robot locomotes with the nominal rolling gait. The end of the pipe is barely visible while the robot moves forward. (d) Sample frames while the robot locomotes with a rectified-head rolling gait. Now the end of the pipe (hence also the pipe joint) is always in the field of view

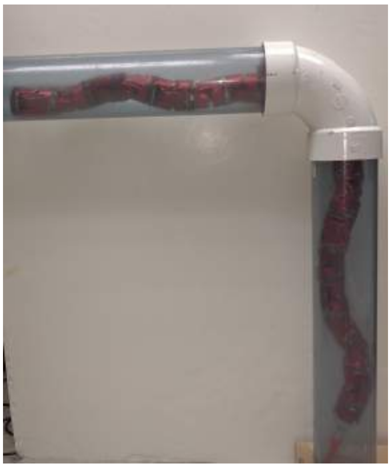


Figure 3.6: Picture of the robot managing a vertical to horizontal 90^o bend

As far as notation is concerned, ${}^{i}T_j$ is the homogeneous transformation from the reference frame i to j, which can be decomposed as

$$ {}^{i}T_j = \begin{bmatrix} {}^{i}R_j & {}^{i}t_j \\ 0^{1\times 3} & 1 \end{bmatrix} \tag{3.1} $$

with ${}^{i}R_j \in \mathbb{R}^{3\times 3}$ being a direction cosine matrix (rotation matrix), ${}^{i}t_j \in \mathbb{R}^{3\times 1}$ a translation vector in metric units.
The relationship among rotation matrices and unitary quaternions are defined as follows:

$$ {}^{i}R_j \equiv {}^{i}q_j \tag{3.2} $$

$$ {}^{i}R_j \cdot {}^{j}R_k \equiv {}^{j}q_k \otimes {}^{i}q_j^{-1} \tag{3.3} $$

$$ {}^{i}R_j \cdot {}^{j}t_k \equiv {}^{i}q_j^{-1} \otimes \begin{bmatrix} 0 \\ {}^{j}t_k \end{bmatrix} \otimes {}^{j}q_k, \tag{3.4} $$

with $\otimes$ denoting quaternion multiplication and q^{-1} denoting quaternion inversion. I append an apostrophe to a variable to denote its value in the previous iteration of the algorithm, e.g. at time step k, $cam = cam(k)$ and $cam' = cam(k-1)$.

Chapter 4

Methods

The goal of this work is to estimate two quantities:

- ${}^{world}T_{cam}(k)$, for $k = 1, \dots, n$ instants of time while the robot is locomoting through the pipe network, i.e. the robot 6-DoF pose
- ${}^{world}T_{pipe_i}$, for the $i = 1, \dots, N$ pipe segments that the robot traversed, i.e. the map of the pipe network.

In order to do it, I devised and implemented the framework shown in Figure 4.1. Each of the following subsections 4.1 to 4.5 describes one of conceptual components of the method, corresponding to the conceptual blocks of the scheme. Major focus is given to the Visual Joint Tracker and the Pipe Estimator, since they are novel to this work. In the explanation of the Complementary Filter and Sensor Fusion methods, I will emphasize how the standard inertial odometry and EKF techniques are customized for the addressed problem.

4.1 Complementary Filter

Orientation estimate

The Complementary Filter is a simple estimation technique which is commonly used to merge the readings from accelerometers and gyroscopes to estimate the orientation of a moving body.

The underlying idea is to low pass filter the estimate of the direction of gravity from the accelerometers and use it to reduce the drift in the orientation estimate obtained by integrating the gyro angular velocity.

In the case of a multi-articulated robots such as snake robots, the high number of IMUs allows for a smart exploitation of the redundant measurements to improve the quality of the orientation estimate.

The extension of the Complementary Filter method from the rigid body case to a multi-body case is straightforward, once the robot kinematics the body is known and the angular velocity of its internal degrees of freedom is compensated for. The multi-body solution used in this work draws inspiration from the work of Rollinson, Buchan and Choset [30], with the difference of using a Complementary rather than an Extended Kalman Filter. The Complementary Filter proved to provide comparable estimation quality while being much less computationally expensive. See [31] for an in-depth comparison between Complementary and Kalman Filter.

The weight among the accelerometer- and gyro-driven estimates is computed according to two elements:

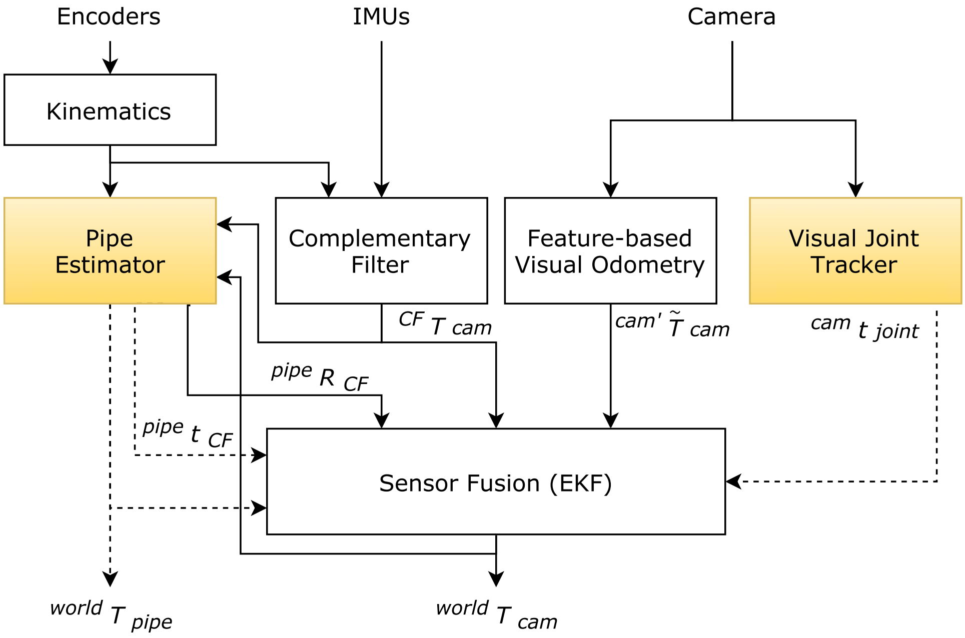


Figure 4.1: High-level schematics of the motion estimation framework. The blocks in yellow are the novel contributions of this work. The dotted lines represent block outputs which may not available at every iteration, e.g. the Visual Joint Tracker can estimate the pipe joint position only when one joint circle is successfully tracked in the current video frame

- the magnitude of the measured linear acceleration: in case the norm of the average acceleration exceeds gravity acceleration more than a given threshold during a certain iteration, the accelerometer orientation estimate is not taken into account, since it implies that forces arising from external contact are affecting the accelerometers
- the magnitude of the measured rotation speed: the higher the norm of the average rotation speed, the more the gyro orientation estimate is taken into account, since the reading is less likely to be driven by noise

It should be highlighted that the Complementary Filter orientation estimate is affected by drift over time in the angle around the gravity vector. This is due to the fact that the gyro-driven estimate is additive, while the accelerometer-driven orientation estimate is absolute but provides information only on two of the three orientation degrees of freedom. This implies that the orientation of the frame CF is not inertial.
The Complementary Filter estimates the orientation of the robot head $^{CF}R_{cam}$. A sample orientation estimate is shown in Figure 4.2.

Position estimate

While in generic applications Complementary Filters only estimate the orientation of the robot, the very structured motion of the robot is exploited to retrieve an estimate of the position of the head in the pipe cross-section plane from the Complementary Filter orientation estimate.
In order to do so, it is assumed that the kinematic center of mass of the robot approximately lies on the pipe main axis. This is a reasonable assumption since the

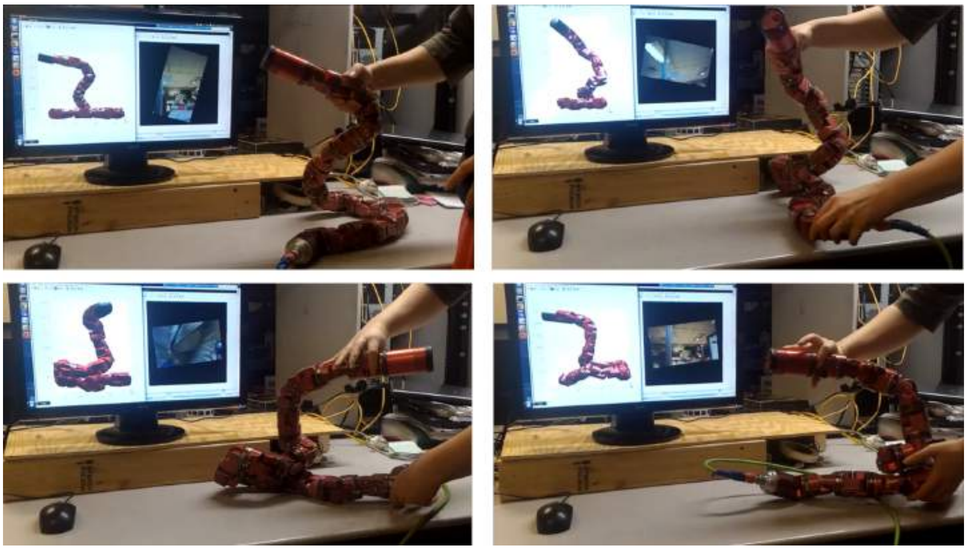


Figure 4.2: Frames from a video sequence of the robot manually rotated of a full turn around the vertical axis. Time flows from left to right, from top to bottom. The monitor behind the robot shows the online estimate of the robot orientation from the Complementary Filter. Over the full turn of the snake, the yaw estimate accumulated an error of around 20^o. The full turn took around 10 seconds

robots propels itself forward by pushing against opposite walls.

Hence, the origin of the Complementary Filter CF is chosen to be at the instantaneous kinematic center of mass of the robot. In this way the position of the head ${}^{CF}t_{cam}$ can be transformed in ${}^{pipe}t_{cam}$ once the relative orientation between the pipe main axis and the Complementary Filter frame is known. The estimates of ${}^{pipe}\begin{bmatrix} y & z \end{bmatrix}^T_{CF}$ are reliable since they are defined by the robot shape and 3D orientation,

These considerations are put into effect in the Sensor Fusion block of this framework as explained in Section 4.5.

As far as inertial position estimate is concerned, the time-integration of the accelerometer readings proved to be very ineffective in estimating the position of the robot. This is due to the kinematics of the snake robot, since each of its rigid body component (and hence IMUs) undergoes a complex motion which cannot be approximated with the overall motion of the robot.

Summary

Summarizing, the Complementary Filter estimates ${}^{CF}T_{cam}$.

The reliability of the its estimate is the following:

- rot($\cdot$, ${}^{world}x$) and rot($\cdot$, ${}^{world}y$) are reliable, while rot($\cdot$, ${}^{world}z$) is prone to drift

- ${}^{pipe}y_{cam}$ and ${}^{pipe}z_{cam}$ are reliable

In order to exploit these considerations, ${}^{world}R_{CF}$ and ${}^{pipe}R_{CF}$ have to be known. Their estimates are provided by other blocks in the proposed framework.

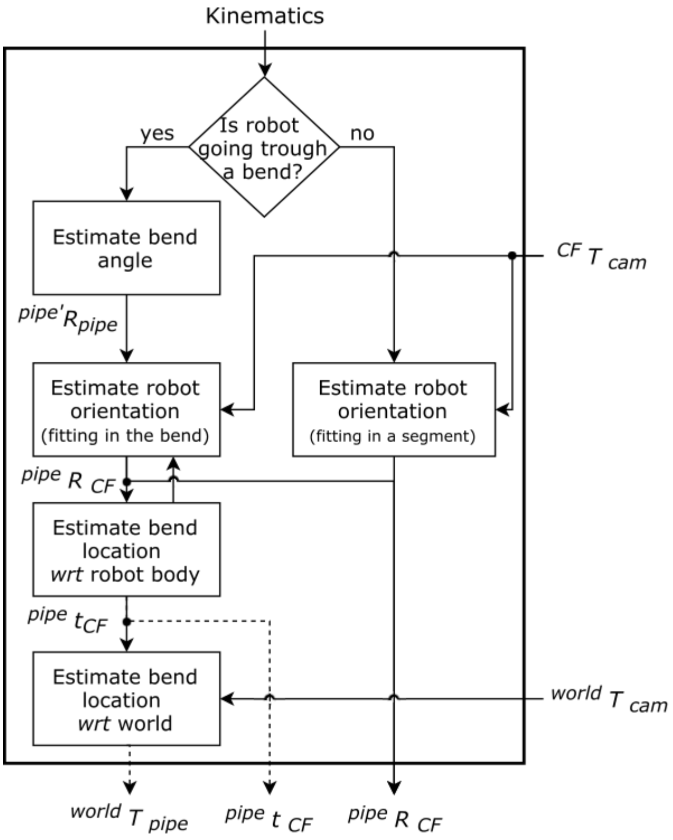

Kinematics
Is robot going trough a bend?
yes
no
Estimate bend angle
$^{pipe'}R_{pipe}$
$^{CF}T_{cam}$
Estimate robot orientation (fitting in the bend)
Estimate robot orientation (fitting in a segment)
$^{pipe}R_{CF}$
Estimate bend location wrt robot body
$^{pipe}t_{CF}$
Estimate bend location wrt world
$^{world}T_{cam}$
$^{world}T_{pipe}$
$^{pipe}t_{CF}$
$^{pipe}R_{CF}$

Figure 4.3: Flowchart of the Pipe Estimator block. The dotted arrows indicate that the output is available only for some iterations of the estimation

4.2 Pipe Estimator

The Pipe Estimator exploits the shape of the robot to compute the drift of the Complementary Filter orientation estimate. In case the robot body is going through a bend at a certain point in time, it also provides an estimate of the three-dimensional position of the robot, and hence of the Complementary Filter frame. Furthermore, the Pipe Estimator is responsible of retrieving the map of the traversed network. This is done by estimating the angles of the bends and the time instant in which the robot transitions from one pipe segment to the next. In order to estimate the length of the pipe segments, the Pipe Estimator is also fed with the current estimate of the pose of the camera in the *world* frame.

The Pipe Estimator algorithm is outlined in Figure 4.3. Each of the conceptual blocks are explained in details in the following subsections 4.2 to 4.2.

For the sake of clarity, the described concepts are shown as applied in a real-world experiment in which the snake robot was crawling through a 45^o bend. Discussion on the resulting estimate is delayed to Chapter 5.3.

Evaluation of pipe geometry

As a first step, the Pipe Estimator evaluates whether the robot is going through a bend in the current time instant. The binary decision is taken according to the shape of the robot by comparing the relative orientation of the two extremities of the robot. Because of the fluctuation due to the locomoting gait, the signals must be filtered for at least the time of one gait cycle. In case the relative orientation of head and tail exceeds a given threshold for a sufficiently long amount of time, the robot is considered to be going through a bend. In the rest of the time, the robot is estimated to be going through a straight pipe segment.

Figure 4.4 shows a sample relative orientation of head and tail over time and how the time is classified by the Pipe Estimator, either as time in which the robot is traversing a bend or going through a straight segment.

Robot orientation in straight pipes

In the time lapses in which the robot is going through a straight pipe segment, the filter estimates the orientation of the Complementary filter frame in the pipe frame, i.e. ${}^{pipe}R_{CF}$. This is done by exploiting the constraint that the robot body is contained inside the pipe, which leads us to assume that time-smoothed main moment of inertia of the robot is aligned with the axis of the pipe. To this end, the principal moments of inertia of the robot taken at its kinematics center of mass is computed through Singular Value Decomposition (SVD) of the zero-mean position of the robot bodies, as explained by Rollinson and Choset in [32].

Bend angle

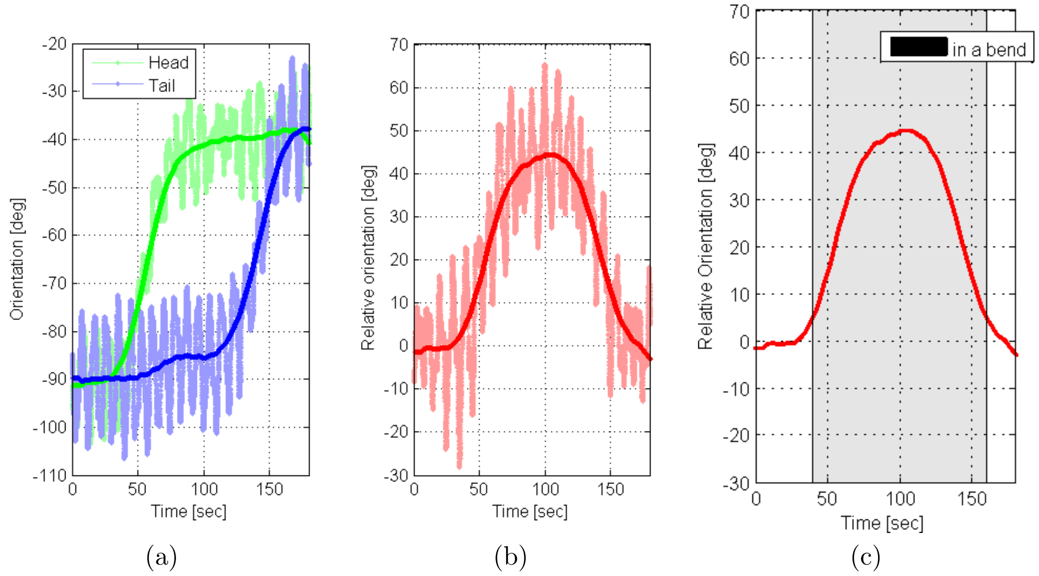
In the time ranges in which the robot is estimated to be going through a bend, the Pipe Filter first estimates the angle of the bend as the peak in magnitude of the relative orientation of the two robot extremities, as shown in Figure 4.5. The rationale behind using the signal peak is that the spacial filtering of the robot shape which is necessary to compute head and tail orientations naturally damps their difference in magnitude. Figure 4.5 shows the estimated pipe angle over time in previous example. The angle of the bend and the plane in which it lies define the relative orientation between the previous and current pipe reference frames, i.e. ${}^{pipe'}R_{pipe}$. The estimated bend angle is used to refine the estimate of the head and tail orientation in the CF reference frame.

Robot pose through bends

Then, the Pipe Estimator estimates the pose of the robot with respect to the two intersecting pipe segments. The estimation is performed by iteratively estimating the location of the bend along the robot body, i.e. ${}^{pipe}t_{CF}$, and the orientation of the robot body inside the bend, i.e. ${}^{pipe}R_{CF}$. The estimation of the bend location along the body is done by intersecting the current estimates of head and tail directions, as shown in Figure 4.6(a). The orientation of the body is computed by using a procedure analogous to the one for the robot inside a straight pipe. The difference is that the principal moment of inertia is computed independently on the two robot parts, separated by the current estimate of the location of the bend along the body. Once the estimates ${}^{pipe}t_{CF}$ and ${}^{pipe}R_{CF}$ have converged to almost constant values, they become outputs of the Pipe Estimator. The refined estimated for the sample experiment used so far are shown in Figure 4.6(b) and (c).

Bend location

Additionally, in the time instant in which the robot head is transitioning from one pipe segment to the next, the Pipe Estimator exploits the current estimate of the pose of the robot ${}^{world}T_{cam}$ from the Sensor Fusion block to update the pipe network



(a) (b) (c)

Figure 4.4: Evaluation of pipe geometry in a sample experiment (a) Orientation of head and tail in the Complementary Filter frame over time in the plane of the bend. Light colors are the non-filtered orientation of head and tail. The oscillations due to the gait cycle are clearly visible. Darker colors are the orientation filtered over time (b) The relative orientation between head and tail. Light red is the non-filtered angle difference, red is the filtered difference (c) Classification of time instants from the Pipe Estimator into time ranges in which the robot is rolling in a straight pipe segment and ranges in which it is traversing a bend. The latter are highlighted in grey

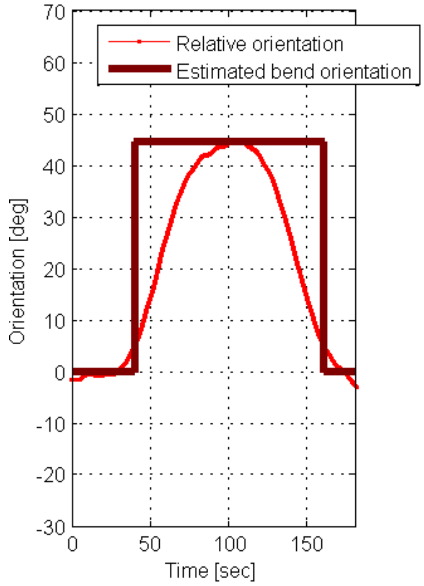
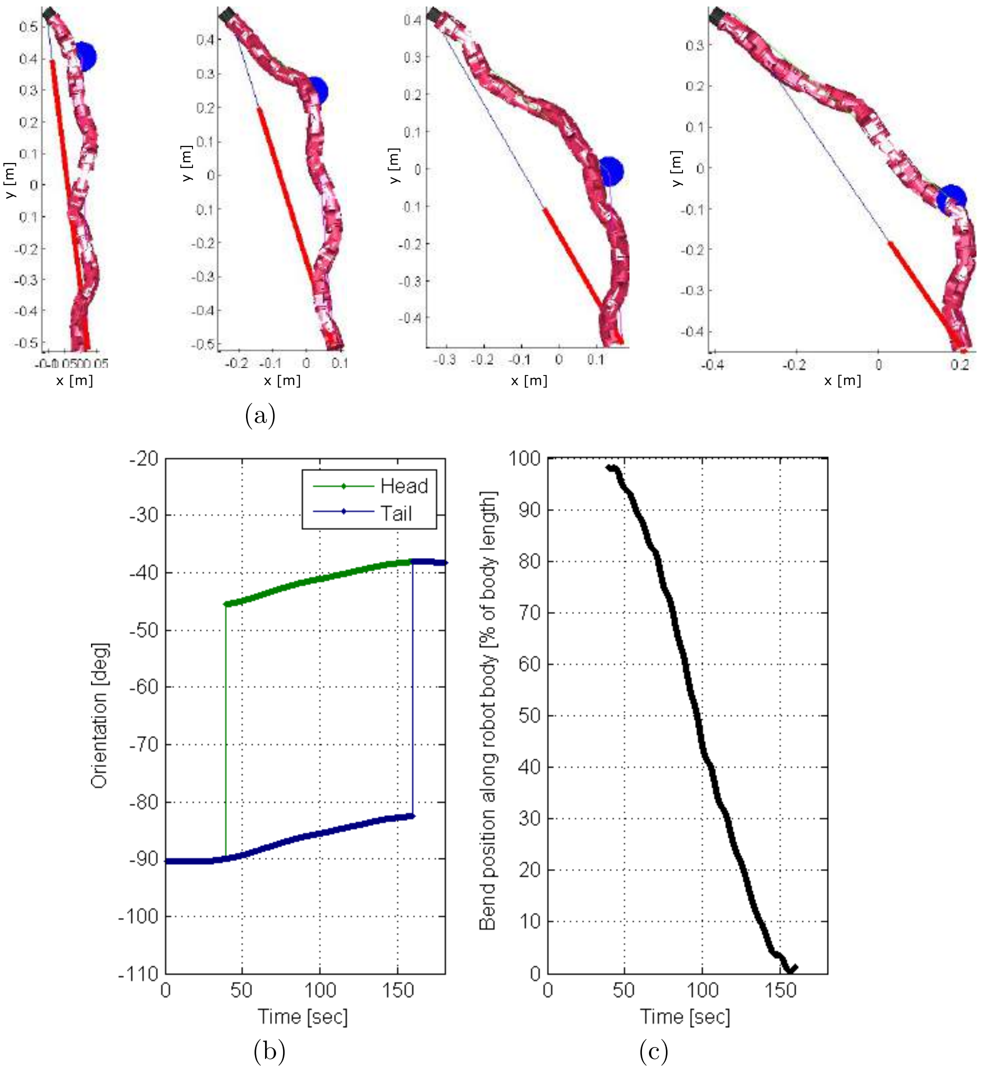


Figure 4.5: Estimate of bend angle as peak of the relative orientation between the robot extremities. Light red is relative orientation as show in Figure 4.4(c). Dark red is the post-processed angle estimate which defines ${}^{pipe'}R_{pipe}$



(a)

(b)

(c)

Figure 4.6: Robot pose through bends. (a) Sequence of robot poses in the Complementary Filter frame with the estimate from the Pipe Estimator of the position of the bend, depicted as a big blue dot. Red is the projection of the bend position on the segment connecting head and tail, which approximates the position of the bend along the robot body. (b) Refined estimate of the head and tail orientation from the Pipe Estimator, respectively in dark green and dark blue. In the time ranges in which the robot is estimated to be in a straight pipe, the two orientations are coincident and coincide with the main axis of inertia of the robot. The head and tail orientation are correlated according to ${}^{pipe'}R_{pipe}$ (c) Evolution over time of the position of the bend along the body in term of body length. It corresponds to the length of the red segment in the frames of sub figure (a) and it is correlated with ${}^{pipe}t_{CF}$

map with the knowledge of the new bend:

$$\begin{bmatrix} {}^{world}t_{pipe} \\ 1 \end{bmatrix} = {}^{world}T_{cam} \cdot {}^{cam}T_{CF} \cdot \begin{bmatrix} {}^{CF}t_{pipe} \\ 1 \end{bmatrix}. \tag{4.1}$$

The orientation of the new pipe segment ${}^{world}R_{pipe}$ is computed from the knowledge of the previous segment orientation and the angle of the connecting bend, i.e.:

$$ {}^{world}R_{pipe} = {}^{world}R_{pipe'} \cdot {}^{pipe'}R_{pipe}. \tag{4.2} $$

The resulting estimate of ${}^{world}T_{pipe}$ is an additional output of the Pipe Estimator.

Summary

To summarize, the Pipe Estimator estimates the following quantities:

- ${}^{pipe}R_{CF}$, that is the drift in orientation of the Complementary Filter
- ${}^{pipe}t_{CF}$, that is the position of the center of mass of the robot. Only available when the robot is traversing a bend, since in this case the robot shape and geometry of the environment entail the robot position
- ${}^{world}T_{pipe}$, that is the orientation of the pipe in which the head of the robot is currently located into as well as the position of the bend. Computed only when the robot transitions form one pipe to the next

4.3 Feature-based Visual Odometry

The current video frame is fed to the Feature-based Visual Odometry algorithm, which matches the salient visual corners with the ones extracted in the video frame of the previous iteration. Epipolar geometry constraints are exploited to retrieve the rotation of the camera between the two frames. The up-to-scale translation of the camera is also computed by triangulating the visual features. The implemented solution follows the Visual Odometry Tutorial of Scaramuzza in [8] and [9]. The details are briefly outlined in the following subsections and a sample algorithm application is shown in Figure 4.7

Feature extraction

At first, salient features are evenly extracted from the current frame.
To avoid the detection of corners at the border between the illuminated and the dark region of the pipe in front robot, the darker areas of the frame are excluded from the corner search. The grey-scale intensity threshold is computed from the Otsu's algorithm [33]. The mask for the feature extraction is made out of the inflated dark regions, in order to exclude the shadow-illumination border.
The extraction is performed following the algorithm from Shi and Tomasi [34]. i.e. using the minimal eigenvalue of gradient matrices as feature quality, and the estimated position of the features is refined at sub-pixel location.
Since the even distribution of the corners is very relevant for the quality of the motion estimate, the frame is divided in smaller region of interest and the extraction of an equal number of features for each region is enforced. To this aim, the minimal accepted quality of the corner is iteratively decreased till the desired number of features is extracted or the value goes below a given threshold. For each extraction, the feature quality is computed as relative to one of the best extracted corner.

Feature matching

In the next iteration of the algorithm, the features locations from the previous video frame are matched to features in the current video frame using the iterative Lucas-Kanade method with pyramids as described in [35]. The unmatched features are discarded, as well as the ones with an error measure much greater than the median error for that iteration. The error measure is the L-1 distance between patches around the original and a moved point, divided by number of pixels in the search window.

Motion estimation

The frame-to-frame feature correspondences are used to estimate the essential matrix using the 5-point algorithm by Nister and Bergen in [6] with RANSAC. Finally, the relative camera rotation and unity-norm translation is retrieved from the essential matrix through chierality check.

The relative scaling between consecutive translations of the camera is computed by exploiting the relative distance of triangulated three dimensional points from the 2D matches. To this aim, features needs to be tracked for at least three frames. The scale is computed as the median of all the relative distances between 3D points, so as to deal with the presence of spurious feature matches.

In future work, the bundle adjustment over camera poses and triangulated 3-D points is a very promising road to take, see e.g. [36].

Summary

The output of this block is the motion of the camera in its local reference frame between the current and previous video frame ${}^{cam'}\tilde{T}_{cam}$. The notation $\tilde{T}$ highlights that the translation estimate is not in metric unit, but up to scale.

4.4 Visual Joint Tracker

The role of the Visual Joint Tracker is to estimate whether a pipe joint is in the field of view of the camera and to return its coordinates in the camera frame in case it is.

Figure 4.8 gives an idea of the appearance of sample pipe joints on which the algorithm has been tested.

Since the visual appearance of joints is a circle, they carry no orientation information. Nevertheless, the virtual motion of the joint in the camera frame mirrors the inertial motion of the camera, and hence the robot.

The work flow of the Visual Joint Tracker is visually represented in Figure 4.9 as well as outlined in pseudo-code in Algorithm 1. The details of each of the conceptual blocks are given in the following subsections 4.4 to 4.4.

In this section, a circle in the video frame is described as $c = \begin{bmatrix} x_{cc} & y_{cc} & r \end{bmatrix}^T$, with (x_{cc}, y_{cc}) being the center of the circle and r its radius, both in pixel units.

Image processing

Pre-processing the video frame greatly improves the performance of the circle detector explained in Section 4.4. A Gaussian filter is used to blur the image in order to smooth away the noise-generated visual features. A Laplacian filter is used to sharpen the edges of the blurred image in order to highlight the contours of the circle generated by the pipe joint. The processed image is a combination of the Gaussian- and Laplacian-filtered images.

Circle detector

A Hough circle detector is run on the processed video frame to detect the most probable circle in the current video frame.

```
Input: Current video frame $image$
Output: If any pipe joint visible:
        Position of joint in $cam$ frame $t_{joint}$ [m]
        Covariance matrix $P_t$ of $t_{joint}$ [m]
$image \leftarrow$ imageProcessing($image$)
$c_{detected} \leftarrow$ houghDetector($image$)
for $i \leftarrow 1$ to $N\_tracks$ do
    $tracks$[i].updatePDAF($c_{detected}$)
    if $tracks[i].P_c > \bar{P}$ then deleteTrack($i$)
    else if $tracks[i].N\_good\_measurements > \bar{m}$ then
        $tracks[i].active \leftarrow true$
end
if $any(tracks.active)$ then
    $is\_joint \leftarrow true$
    $j \leftarrow$ chooseTrack($tracks.active$)
    $c_{filt} \leftarrow tracks[j].x$; $P_{c_{filt}} \leftarrow tracks[j].P_c$
    $t_{joint} \leftarrow$ pinholeModel($c_{filt}$)
    $P_t \leftarrow$ propagateP(@pinholeModel,$P_c$,$c_{filt}$)
end
```

Algorithm 1: Visual Joint Tracker pseudo-code

The Hough transform is a feature extraction technique which relies on a voting procedure. In the Hough gradient method, pixels vote for the circles which they are likely to lie on the circumference of. The vote depends on the image gradient in the surrounding image patch. The voting pixels are selected with a Canny edge detector as in [38].
The votes are stored in a three-dimensional accumulator, whose dimensions correspond to the circle parameters x_{cc}, y_{cc} and r.
The coordinates of the global maximum in the voting space are the parameters defining the detected circle. See [39] for details on the procedure.
Let us call the circle extracted by the Hough detector as $c_{detected}$.

Tuning of image processing and circle detector parameters

The performance of the circle detector largely depends on the parameters used in the image preprocessing and in the Hough transformation.
While a trial-and-error tuning is possible, I devised an automatic selection procedure which can be used every time a different pipe network is explored. The chosen parameters are expected to be optimal for pipes and joints which look similar to the one used in the tuning procedure.
In details, the parameters which proved to be more relevant for the algorithm performance are:

- the radius of the Gaussian filter
- the resolution of the Hough voting space
- the threshold for the Canny detector hysteresis procedure

The optimization problem relies on ground truth about the position and size of the joint circles at some sample frames of the video used in the tuning procedures. This can be achieved by manually annotating the circle parameters for some frames in which the pipe joint is in the field of view. A dense search is performed across the

parameter space by running the image processing and the Hough detector steps on the video frames for which the ground truth is known. The resulting detected circles are compared to the ground truth circles and the parameter combination cost is increased according to their discrepancy, measured as normalized non-overlapping circle area. The minimum cost parameters are chosen.
While there is no formal guarantee on the convexity of the mentioned optimization problem, empirical experiments showed that one local minimum can be usually found.

Circle tracker

The circle $c_{detected}$ is the input of a circle tracker, whose role is to evaluate whether there is a pipe-generated circle in the current video frame and to provide its refined size and position coordinates in case there is.
To this end, the circle tracker carries several candidate tracks (or hypothesis or observers) of the current state of the pipe-generated circle. The state of the circle is $\xi_c = \begin{bmatrix} c \\ \dot{c} \end{bmatrix} \in \mathbb{R}^{6\times 1}$ and it is modeled as a linear time-varying system with Gaussian noise in its process and measurement models.
Each observer is a Probabilistic Data Association Filter [40]. PDAFs are a variant of Kalman Filter which additionally provide a probabilistic estimate on whether the measurement is clutter- or target-generated. This additional discrimination is of great benefit in the considered setup, since the Hough detector is likely to extract circles in the shadow region generated by the self-carried light source. For an in-depth discussion of PDAF the author recommends the Doctoral Thesis of Rodningsby [41]. Figure 4.10 shows a comparison between a Kalman Filter and a Probabilistic Data Association Filter of the pipe-joint circle in a sample video.
At each iteration of the circle tracer, the circle $c_{detected}$ is used to start a new candidate track, whose initial state is $\xi_c(0) = \begin{bmatrix} c_{detected} & 0^{3\times 1} \end{bmatrix})$ and to update the existing candidate tracks as measurement $\zeta_c(k) = c_{detected}$.
A constant velocity model is used for the circle state motion:

$$\xi_c(k) = A_c \cdot \xi_c(k-1) + w_{c,x} = \begin{bmatrix} 1&0&0&1&0&0 \\ 0&1&0&0&1&0 \\ 0&0&1&0&0&1 \\ 0&0&0&1&0&0 \\ 0&0&0&0&1&0 \\ 0&0&0&0&0&1 \end{bmatrix} \cdot \xi_c(k-1) + w_{c,x}. \tag{4.3}$$

The measurement model is the following:

$$\zeta_c(k) = C_c \cdot \xi_c(k) + w_{c,\zeta} = \begin{bmatrix} 1&0&0&0&0&0 \\ 0&1&0&0&0&0 \\ 0&0&1&0&0&0 \end{bmatrix} \cdot \xi_c(k) + w_{c,\zeta}, \tag{4.4}$$

where the noises $w_{c,x}$ and $w_{c,z}$ are mutually independent and Gaussian distributed, whose covariance matrices are respectively Q_c and R_c.
The prediction step as well as the innovation and the filter gain are identical to standard Kalman Filter ones:

$$\xi_c(k|k-1) = A_c \cdot \xi_c(k-1|k-1) \tag{4.5}$$

$$P_c(k|k-1) = A_c \cdot P_c(k-1|k-1) \cdot A_c + Q_c \tag{4.6}$$

$$v_c(k) = \zeta_c(k) - C_c \cdot \xi_c(k|k-1) \tag{4.7}$$

$$S_c(k) = C_c \cdot P_c(k|k-1) \cdot C_c^T + R_c \tag{4.8}$$

$$K_c(k) = P_c(k|k-1) \cdot C_c^T \cdot S_c^{-1}. \tag{4.9}$$

The PDAF update step differs from the Kalman Filter one due to the 'association probability' β. The parameter β represents the probability that the current measurement is target-generated rather than spurious and it is computed as

$$\beta = k \cdot e^{-\frac{1}{2} \cdot v_c(k) \cdot S_c(k)^{-1} \cdot v_c(k)}, \tag{4.10}$$

with k being a normalizing constant which depends on the real target detection probability P_D, the gate probability P_G, the innovation matrix $S_c(k)$ and the dimension of the measurement vector z_c. Since there is only one measurement per iteration, the 'spread of innovation' matrix is null.
Hence, the updated state and its covariance are

$$\xi_c(k|k) = \xi_c(k|k-1) + \beta(k) \cdot K_c(k) \cdot v_c(k) \tag{4.11}$$

$$P_c(k|k) = P_c(k|k-1) - \beta(k) \cdot K_c(k) \cdot v_c(k) \cdot K_c(k)^T. \tag{4.12}$$

After all the PDAFs are updated, the tracks whose norm of the a-posteriori covariance exceed a defined threshold are discarded. Furthermore, when the similarity between two tracks exceeds a given threshold, one of the two is discarded. The difference $d_{i,j}$ among tracks i and j is computed as their Bhattacharya distance:

$$d_{i,j} = \frac{1}{8}(\xi_{c,i}(k|k) - \xi_{c,j}(k|k))^T \cdot P_{c,i,j}^{-1}(\xi_{c,i}(k|k) - \xi_{c,j}(k|k)) + \tag{4.13}$$

$$+ \frac{1}{2} ln(\frac{det(P_{c,i,j})}{\sqrt{det(P_{c,i}(k|k)) \cdot det(P_{c,i}(k|k))}}), \tag{4.14}$$

with $P_{c,i,j} = \frac{P_{c,i}(k|k)+P_{c,j}(k|k)}{2}$.
If none of the surviving tracks has collected enough validating measurements ($\beta(k) > 50\%$) in a given number of previous iterations, the circle tracker evaluates that no joint is visible in the current video frame.
Otherwise, a track is chosen among the candidates, according to its a-posteriori covariance as well as validation history. The position and size of the a-posteriori circle $c_{filt} = \xi_c(1:3)$ and its covariance $P_{c_{filt}} = P_c(1:3, 1:3)$ are propagated.

Pinhole camera model

Finally, c_{filt} is transformed into metric coordinates of the pipe joint using a pinhole camera approximation and the perspective projection equation:

$$^{cam}t_{joint} = f_{pinhole}(c_{filt}) = \begin{bmatrix} \frac{(x_{cc}-x_{pp}) \cdot r}{r} & \frac{(y_{cc}-y_{pp}) \cdot r}{r} & \frac{f \cdot r_{real}}{r} \end{bmatrix}^T, \tag{4.15}$$

with f being the focal length of the camera, (x_{pp}, y_{pp}) the camera principal point, both in pixel units and r_{real} the real radius of the pipe in metric units. The size of the pipes can be either required as an input of the framework or estimated from the shape of the robot. The online estimate of the pipe radius can be performed following the work of Enner, Rollinson and Choset in [3].
One of the advantages of using PDAFs for the tracking of joint-generated circle is that the c_{filt} comes with its uncertainty P_c, which can be used to compute the uncertainty on $^{cam}t_{joint}$ from uncertainty propagation rules:

$$P_{joint} = \left.\frac{\partial f_{pinhole}}{\partial c}\right|_{c_{filt}} \cdot P_{c_{filt}} \cdot \left.\frac{\partial f_{pinhole}}{\partial c}\right|_{c_{filt}}^T. \tag{4.16}$$

In case the pipe radius is online estimated, the uncertainty of the r_{real} estimate should be included in the computation of P_{joint}.

Summary

The output of this block is the position of the next visible joint in the current camera frame ${}^{cam}t_{joint}$ together with its associated uncertainty P_{joint}.

In order to exploit this information, it should be noted that the relative position between the camera frame and the pipe joint is an affine function of the relative position between the camera frame and the current pipe frame. In symbols, ${}^{pipe}R_{cam} \cdot {}^{cam}t_{joint}$ is an affine function of ${}^{pipe}t_{cam}$. This relationship is exploited in the Sensor Fusion block.

It is important to highlight that the z-axis of the camera is approximately aligned with the x-axis of the pipe frame since the gait of the robot is such that the camera is mostly facing forward. It follows that ${}^{cam}z_{joint} \approx {}^{pipe}x_{cam}$. This consideration entails that the absolute forward motion of the robot is directly estimated by the Visual Joint Tracker, without any further contamination from other estimates. This characteristic makes the Visual Joint Tracker a very important element in the given framework, as it will be further discussed in Chapter 5.3.

The drawback of the Visual Joint Tracker is that there may be points in time in which no joint circle is in the camera field of view. This depends on setup parameters such as the power of the light source, the length of the pipes and the oscillations with respect to the forward direction that the camera undergoes during the robot gait.

4.5 Sensor fusion

The Sensor Fusion block has two roles:

- combine the estimates from the previously explained estimators in order to retrieve observations of the EKF state (Section 4.5.1)
- run an EKF aimed at estimating the pose of the robot over time as well as the pipe network map (Section 4.5.2)

At every iteration, the Extended Kalman Filter provides an estimate of the following quantities:

- the pose of the camera ${}^{world}T_{cam}$
- the pose of the current pipe ${}^{world}T_{CF}$
- the pose of the camera in the previous iteration of the filter ${}^{world}T_{cam'}$
- the pose of the Complementary Filter frame in the current pipe ${}^{pipe}T_{CF}$

The first two quantities are the goal of the framework. The second two are tracked over time to allow the observation of the first two from the available measurements. For sake of clarity and consistency with the previous discussion, the pose estimates are described in homogeneous transformation notation. The quaternion-translation notation used in the EKF is introduced only in Subsection 4.5.2. For this reason the EKF measurements expressed as homogeneous transformations are called $\check{\zeta}$, while the measurements using quaternion notation are defined as ζ.

4.5.1 Estimates to EKF measurements

Complementary Filter estimate

The Complementary Filter estimates the pose of the camera in the Complementary Filter frame ${}^{CF}T_{cam}$ and it is a direct observation of the EKF state, since

$$ {}^{CF}T_{cam} = {}^{CF}T_{pipe} \cdot {}^{pipe}T_{world} \cdot {}^{world}T_{cam}. \tag{4.17} $$

Hence the EKF measurement $\check{\zeta}_{CF}$ is defined as

$$\check{\zeta}_{CF} = {}^{CF}T_{cam}. \tag{4.18}$$

The uncertainty of the measurement depends only on the Complementary Filter estimate accuracy.

Feature-based Visual Odometry estimate

The estimate of the Feature-Based Visual Odometry is the local rotation of the camera between two frames and the up-to-scale translation ${}^{cam'}\tilde{T}_{cam}$. In order to use it as an observation of the EKF state, the scale of the translation must be resolved.

To this end, the translation estimate from the Complementary Filter is exploited. I discussed in Section 4.1 how the Complementary Filter is able to estimate the position of the camera in the pipe cross-section plane. The previously-mentioned assumption that the camera is mostly facing forward, ${}^{cam}z \approx {}^{pipe}x$, implies that ${}^{pipe}\begin{bmatrix} y & z \end{bmatrix}_{cam} \approx {}^{cam'}\begin{bmatrix} x & y \end{bmatrix}_{cam}$. Hence the Complementary Filter is able to provide an estimate of the relative translation of the camera in the camera plane $\Delta^{cam'}\begin{bmatrix} x & y \end{bmatrix}_{cam}$ by exploiting two consecutive estimates of ${}^{CF}T_{cam}$, as follows:

$${}^{cam'}T_{cam} = {}^{cam'}T_{CF} \cdot {}^{CF}T_{cam} \tag{4.19}$$

$$\Delta({}^{cam'}\begin{bmatrix} x & y \end{bmatrix}_{cam}) = {}^{cam'}T_{cam}(1:2,4). \tag{4.20}$$

The metric unit translation from the Complementary Filter is compared to the up-to-scale translation from the Feature-Based Visual Odometry estimate and the scale λ is computed as:

$$\lambda = \frac{\Delta({}^{cam'}\begin{bmatrix} x & y \end{bmatrix}_{cam,CF})}{\Delta({}^{cam'}\begin{bmatrix} x & y \end{bmatrix}_{cam,VO})}. \tag{4.21}$$

The measurement from the Feature-Based Visual Odometry $\check{z}_{VO}$ is hence defined as

$$\check{\zeta}_{VO} ={}^{cam'} T_{cam} = \begin{bmatrix} {}^{cam'}R_{cam} & {}^{cam'}\tilde{t}_{cam} \cdot \lambda \\ 0^{1\times 3} & 1 \end{bmatrix}. \tag{4.22}$$

Its uncertainty depends on the Feature Based Visual Odometry estimate accuracy and the Complementary Filter estimate one.

The measurement is a function of the EKF state according to:

$$\check{\zeta}_{VO} ={}^{cam'} T_{world} \cdot{}^{world} T_{cam}. \tag{4.23}$$

Visual Joint Tracker

The estimate of the Visual Joint Tracker is the position of the next pipe joint in the camera frame ${}^{cam}t_{joint}$. The estimate is available only when its internal circle tracker estimates that a pipe joint is in the camera field of view.

In order to use ${}^{cam}t_{joint}$ as an observation of the EKF filter state, it should be exploited the affinity function between the position of the camera with respect to the joint and the position of the camera in the current pipe frame, i.e. ${}^{pipe}t_{(joint-cam)}$ and ${}^{pipe}t_{cam}$, which was discussed in Section 4.4. In order to compute ${}^{pipe}t_{(joint-cam)} = {}^{pipe}R_{cam} \cdot {}^{cam}t_{joint}$ from the Visual Joint Tracker estimate, the orientation ${}^{pipe}R_{cam}$ is needed. It is retrieved from the Complementary Filter estimate of ${}^{CF}R_{cam}$ and the Pipe Estimator estimate of ${}^{pipe}R_{CF}$. Finally, since ${}^{pipe}t_{(joint-cam)}$ is an affine function of the EKF state, $\Delta({}^{pipe}t_{(joint-cam)}) =$

$\mathit{\Delta}({}^{pipe}t_{cam})$. Hence, by taking difference among two subsequent estimates of the ${}^{pipe}t_{(joint-cam)}$, we obtain an observation of the state:

$$\check{\zeta}_{VJT} = \mathit{\Delta}({}^{pipe}t_{(joint-cam)}) = {}^{pipe}t_{(joint-cam)} - {}^{pipe'}t_{(joint-cam)'}. \tag{4.24}$$

In other words, the measurement $\check{\zeta}$ it is the translation of the camera with respect to the next pipe joint, as described in the current pipe coordinate. The translation of the camera with respect to the next pipe joint mirrors the translation of the camera in the current pipe frame. Hence, the measurement $\check{\zeta}_{VJT}$ is a function of the EKF state according to:

$$\check{\zeta}_{VJT} = -\mathit{\Delta}({}^{pipe}t_{cam}) = {}^{pipe}T_{world} \cdot {}^{world}t_{cam'} - {}^{pipe}T_{world} \cdot {}^{world}t_{cam}, \tag{4.25}$$

where I used the approximation ${}^{pipe'}T_{world} \approx {}^{pipe}T_{world}$. This is a reasonable since the only instants of times in which ${}^{world}T_{pipe}$ differs from ${}^{world}T_{pipe'}$ is when the robot transitions from one pipe segment to the next. A part from being a very little amount of the whole experiment time, the Visual Joint Tracker is unlikely to be tracking a pipe joint in these instants, due to the abrupt change of the camera field of view.

Summarizing, the measurement $\check{\zeta}_{VJT}$ is created assembling estimates from the Visual Joint Tracker, the Complementary Filter and the Pipe Estimator Block over two time instants:

$$\check{\zeta}_{VJT} = {}^{pipe}R_{CF} \cdot {}^{CF}R_{cam} \cdot {}^{cam}t_{joint} - {}^{pipe'}R_{CF'} \cdot {}^{CF'}R_{cam'} \cdot {}^{cam'}t_{joint'}. \tag{4.26}$$

The uncertainty of the measurement observation is hence affected by the uncertainty of three estimates. Nevertheless, the considerations presented in Section 4.4 should be exploited: since the ${}^{cam}z_{joint} \approx {}^{pipe}x_{cam}$, the estimate of $\mathit{\Delta}({}^{pipe}x_{(joint-cam)})$ can be approximately considered as dependent only on the Visual Joint Tracker. This entails that its reliability is greater than the other two translation components.

Pipe Estimator

The estimates from the Pipe Estimator are functions of the EKF state, hence the Pipe estimator measurements are defined as

$$\check{\zeta}_{PE,R_{CF}} = {}^{pipe}R_{CF} \tag{4.27}$$

$$\check{\zeta}_{PE,t_{CF}} = {}^{pipe}t_{CF} \tag{4.28}$$

$$\check{\zeta}_{PE,R_{pipe}} = {}^{world}R_{pipe} \tag{4.29}$$

$$\check{\zeta}_{PE,t_{pipe}} = {}^{world}t_{pipe}. \tag{4.30}$$

As it was discussed in Section 4.2, the estimate ${}^{world}t_{pipe}$ is only available when the robot transitions from one pipe segment to the next and the estimate ${}^{pipe}t_{CF}$ is not available when the robot is inside a straight pipe segment. In the latter case, the previously-discussed approximation is used: ${}^{pipe}\begin{bmatrix} y & z \end{bmatrix}_{CF} \approx \begin{bmatrix} 0 & 0 \end{bmatrix}$ and hence only $\check{\zeta}_{PE,t_{pipe}}(1:2)$ is available.

It should be highlighted that the measurements $\check{\zeta}_{PE,R_{CF}}, \check{\zeta}_{PE,t_{CF}}$ and $\check{\zeta}_{PE,R_{pipe}}$ were generated from the Complementary Filter estimates and $\check{\zeta}_{PE,t_{pipe}}$ depends also on the current state estimate of the EKF. Since these dependencies cannot be straightforwardly included in the measurement model at the current stage of the work they are neglected.

4.5.2 EKF

In this subsection, the formulation of the Extended Kalman Filter is explained. The relationships explained in Section 3.3 are exploited to convert the pose from homogeneous transformation to a vector made of a unitary quaternion and a translation vector. Hence, the measurement introduced in the previous section as $\tilde{\zeta} = {}^{i}T_j \in \mathbb{R}^{4\times4}$ are converted to $\zeta = {}^{i}(q,t)_j \in \mathbb{R}^{7\times1}$.
The EKF state vector is:

$$\xi(k) = \begin{bmatrix} {}^{world}(q,t)_{cam} & {}^{world}(q,t)_{cam'} & {}^{world}(q,t)_{pipe} & {}^{pipe}(q,t)_{CF} \end{bmatrix}^T \in \mathbb{R}^{28\times1}, \tag{4.31}$$

with covariance matrix $P \in \mathbb{R}^{28\times28}$.
The Filter is initialized according to the following considerations:

- The camera and previous camera pose are initialized according to the Pipe Estimator and Complementary Filter readings:

$$^{world}(q,t)_{cam}(0) =^{world} (q,t)_{cam'}(0) = f(z_{CF}(0), z_{PE,R_{CF}}(0), z_{PE,t_{CF}}(0)) \tag{4.32}$$

- The first pipe segment is considered as aligned with the world frame:

$$^{world}(q,t)_{pipe}(0) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^T \tag{4.33}$$

- The initial pose of the Complementary Filter is chosen to coincide with the first Pipe Estimator estimate:

$$^{pipe}(q,t)_{CF}(0) = \begin{bmatrix} \zeta_{PE,R_{CF}}(0) & \zeta_{PE,t_{CF}}(0)) \end{bmatrix}^T . \tag{4.34}$$

As far as the prediction step is concerned, the camera pose is propagated forward with a visual-odometry data-driven model, whose function is denoted as $g(\cdot)$. Since $\check{\zeta}_{VO} = {}^{cam}T_{world}(k-1) \cdot^{world} T_{cam}(k)$ it follows that: ${}^{world}(q,t)_{cam}(k) = g({}^{world}(q,t)_{cam}(k-1), \zeta_{VO}(k))$. A zero-velocity motion model is used for the the other components of the state:

$$^{world}(q,t)_{cam'}(k) = {}^{world}(q,t)_{cam}(k-1) \tag{4.35}$$

$$^{world}(q,t)_{pipe}(k) = {}^{world}(q,t)_{pipe}(k-1) \tag{4.36}$$

$$^{pipe}(q,t)_{CF}(k) = {}^{pipe}(q,t)_{CF}(k-1). \tag{4.37}$$

As far as the measurement update step is concerned, the full measurement vector is:

$$\zeta = \begin{bmatrix} \zeta_{CF} & \zeta_{VJT} & \zeta_{PE,R_{CF}} & \zeta_{PE,t_{CF}} & \zeta_{PE,R_{pipe}} & \zeta_{PE,t_{pipe}} \end{bmatrix}^T \in \mathbb{R}^{24\times1}. \tag{4.38}$$

The update step is performed at the rate of the slowest sensor modality. As it was discussed several time so far, not all the measurements are available at every iteration of the EKF. In particular:

- If no pipe joint is currently visually tracked, $\zeta_{VJT} \in \mathbb{R}^{3\times1}$ is not available
- If the robot is not going through a bend, $\zeta_{PE,t_{CF}} \in \mathbb{R}^{3\times1}$ is not available, but it is replaced the pipe-cross-section position estimate $\zeta_{PE,[yz]_{CF}} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \in \mathbb{R}^{2\times1}$
- If the robot is not transitioning from a pipe segment to the next, $\zeta_{PE,t_{pipe}} \in \mathbb{R}^{3\times1}$ is not available.

The measurement model of the full measurement vector is

$$\zeta = h(\xi(k)) + w_\zeta \tag{4.39}$$

with the measurement function $h(\cdot)$ being derived by converting the homogeneous-matrix relationship explained in Section 4.5.1 into quaternion-matrix notation, as defined in Section 3.3. The noise w_ζ is assumed to be white Gaussian noise with covariance matrix $R \in \mathbb{R}^{28\times 28}$.

The linearized system is observable in the iterations in which all the measurements are available.

Finally, standard EKF equations are used to propagate the state:

$$\xi(k|k-1) = f(\xi(k-1|k-1) = \tag{4.40}$$

$$P(k|k-1) = F(k-1) \cdot P(k-1|k-1) \cdot F(k-1)^T + Q \tag{4.41}$$

$$v(k) = \zeta(k) - h(\xi(k|k-1)) \tag{4.42}$$

$$S(k) = H(k) \cdot P(k|k-1) \cdot H(k)^T + R \tag{4.43}$$

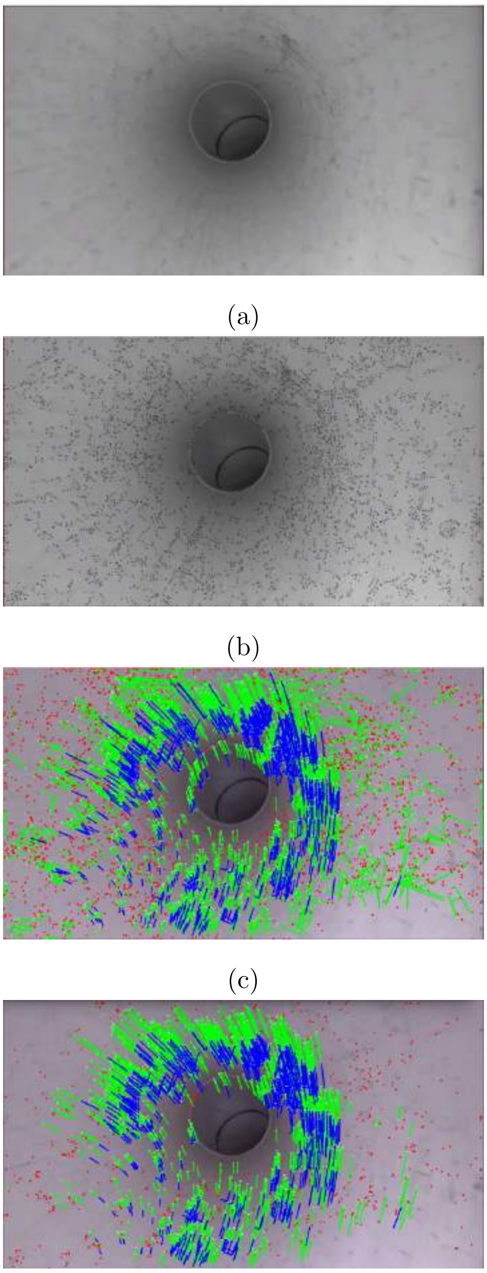
$$K(k) = P(k|k-1) \cdot H^T \cdot S(k)^{-1} \tag{4.44}$$

$$\xi(k|k) = \xi(k|k-1) + K(k) \cdot v(k) \tag{4.45}$$

$$P(k|k) = (I^{28\times 28} - K(k) \cdot H) \cdot P(k|k-1), \tag{4.46}$$

with $F(k-1) = \frac{\partial f}{\partial \xi(k-1)}\Big|_{\xi(k-1|k-1)}$ and $H(k) = \frac{\partial h}{\partial \xi(k)}\Big|_{\xi(k|k-1)}$.

The residual is computed as a normal subtraction of the predicted measurement from the actual one, as in the tutorial of Technische Universitaet Muenchen in [42]. A more accurate solution would involve using an indirect Kalman Filter and using small quaternion-error approximation as in e.g. [10].



(a)

(b)

(c)

(d)

Figure 4.7: Sample performance of the Feature Based Visual Odometry (a) Original video frame (b) Black circles are extracted visual features (c) Red circles are unmatched features, green lines connect the current position and the previous position of matched features, blue lines are the history of the positions of the matched features (d) Red circles are features which are considered outliers in the RANSAC-based essential matrix estimation or which do not survive the chierality check

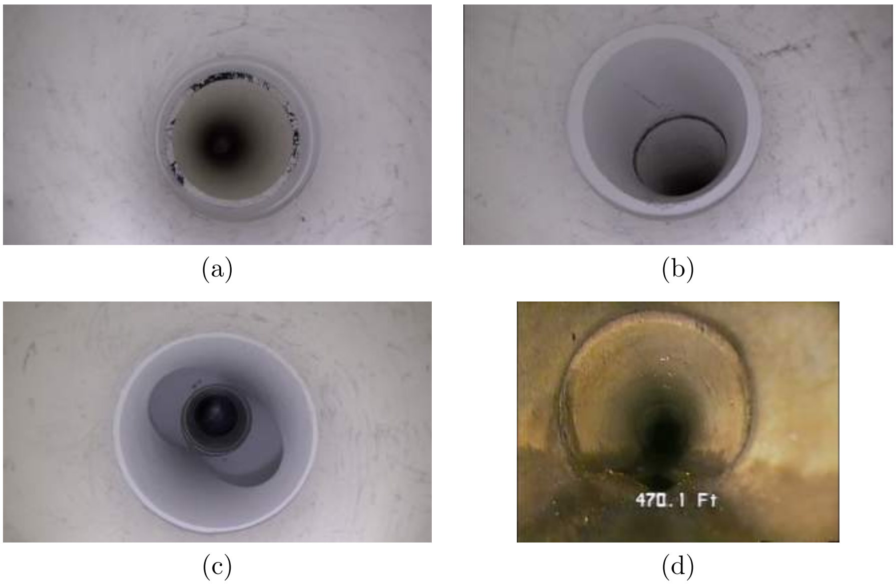

(a) (b) (c) (d)

Figure 4.8: Pipe joints as seen from the frontal camera (a) Joint between a pipe and a straight junction (b) Joint between a pipe and a 90^o bend (c) Joint between a pipe and a fork (d) Joint between two sealed pipes (figure from [37])

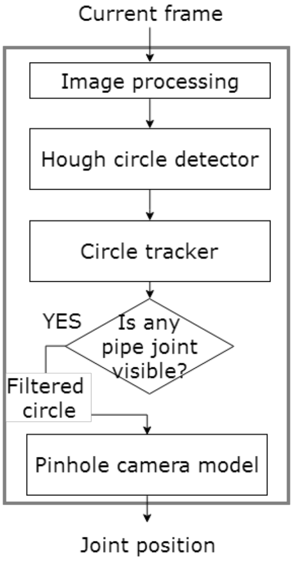
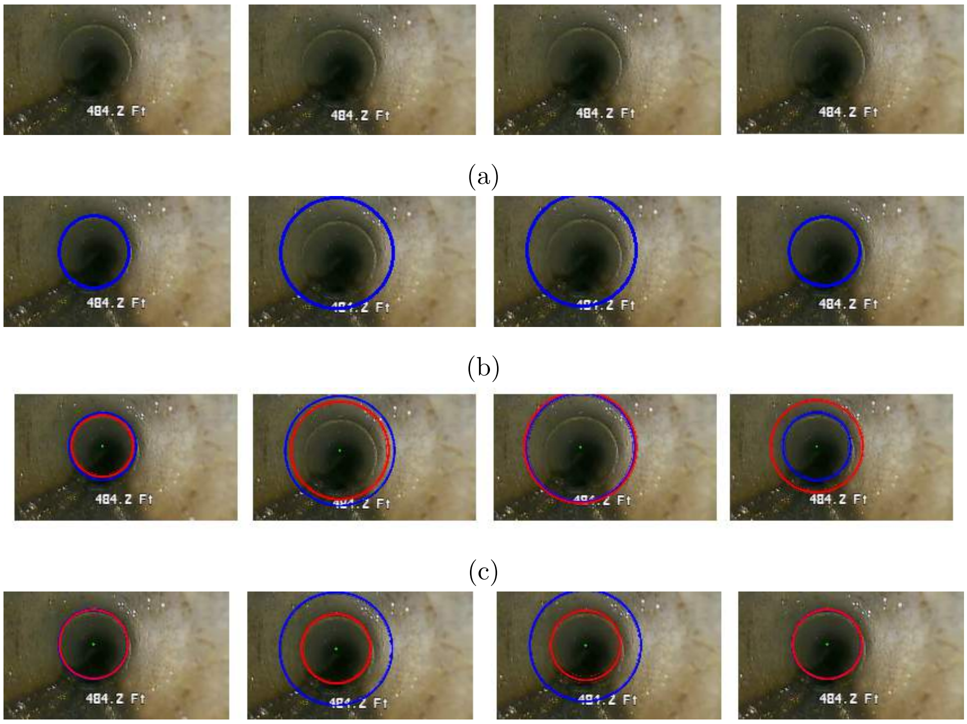


Figure 4.9: Visual Joint Tracker overview



(a)

(b)

(c)

(d)

Figure 4.10: Comparison between Kalman Filter and Probabilistic Data Association Filter (a) Original consecutive video frames (video from [37]). In all of them the pipe joint is in the field of view (b) Blue is the circle detected by the Hough detector. The detection at the second and third time instant are spurious, i.e. they don't represent the pipe-joint (c) Red is the a-posteriori estimate of the joint-generated circle according to the Kalman Filter. In the first time instant, the pipe joint is properly tracked, while in the second and third time instants the noise-generated detected circle drives the estimate away. (d) Red is the a-posteriori estimate of the joint-generated circle according to the Probabilistic Data Association Filter. In the second and third time instants, the detected circle is classified as spurious ($\beta \approx 0$) and the a-priori state is not updated. This leads to a superior tracking performance with respect to the Kalman Filter.

Chapter 5

Results and discussion

This chapter presents some results of the proposed algorithm in a live scenario as well as their interpretation. Section 5.1 provides details on the empirical setup. Sections 5.2 to 5.2 show the results of each of the individual motion estimators. Section 5.2 presents some results of the reconstructed pipe map and the estimate of the robot motion, which are the ultimate output of the proposed framework. Section 5.3 discusses the mentioned results.

5.1 Setup

The proposed estimation framework is implemented as follows:

1. The snake robot is position-controlled in a rolling gait with rectified head through a pipe network. MATLAB is used to send motion commands as well as to receive feedback from the robot encoders and IMUs. The sensor readings are fed to an on-line MATLAB implementation of the Complementary Filter. The video from the frontal camera is recorded.
2. The video is fed to the Feature-based Visual Odometry and the Visual Joint Tracker estimators. I implemented them in a C++ software package, which makes use of the open source Computer vision library OpenCV [43]
3. The Complementary Filter estimate is fed to the Pipe Estimator, the individual estimates are merged together and the Extended Kalman Filter is run in a MATLAB software package

For an extensive documentation on the implementation, the author refers to the Wiki as well as the code which she made available on [44].
The proposed estimation framework is tested on a snake robot with 18 modules, whose proprioceptive sensors were sending readings at a frequency of 100 Hz.
The snake is moving at 5 to 10 seconds per full gait cycle. The average forward motion depends on the amount of slipping that the robot undergoes during each experiment.
The used monocular frontal camera is a GoPro Hero 4 Session [45]. It was suitable for the purpose of this algorithm because of its wide field of view ($\approx 150^o$ [46]) as well as fixed focal length, which is a requirement of the visual odometry part of the proposed solution. The intrinsic parameters of the GoPro have been computed using the camera calibration tools provided by OpenCV. Depending on the snake gait speed, the video is processed at either 6 or 15 FPS.
The used pipe network consists of pipes of 0.10 m in diameter, with bends of 45^o, 90^o and 180^o. The tested pipe networks are horizontal.

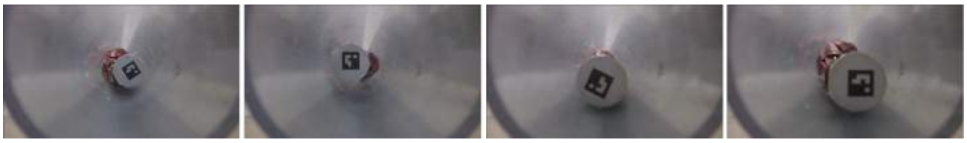


Figure 5.1: Frames from the video used to track the April Tag mounted on the snake head while it is crawling through a horizontal pipe

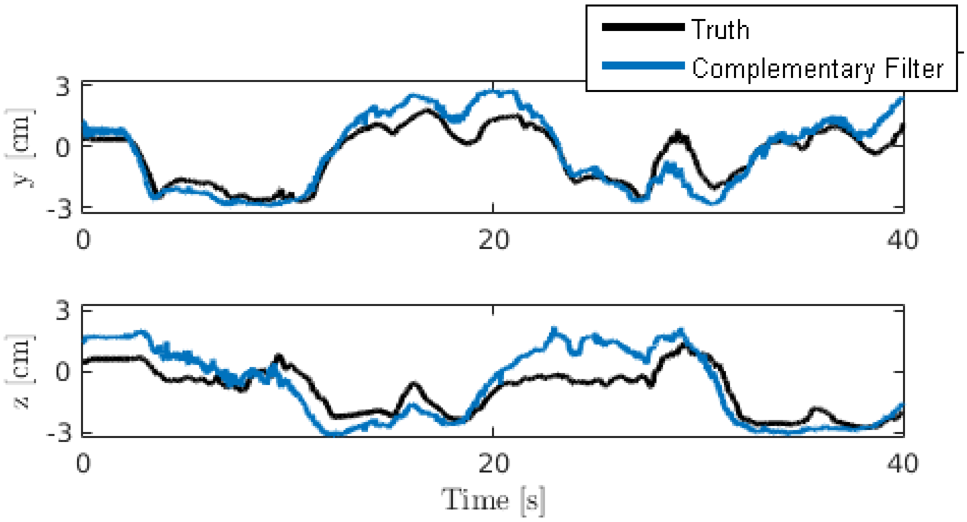


Figure 5.2: Head position in the cross section plane of the pipe. Black is ground truth, blue is the Complementary Filter estimate of ${}^{pipe}t_{cam}$. Horizontal direction is y, vertical direction is z. RMSE is 1 cm

For the sake of simplicity, the covariance matrices of the EKF measurements are assumed diagonal, neglecting the correlation among the measurements. Future work should address this simplification, so as to test the sensor fusion performance when exploiting the correlation among the given signals. This includes computing the full measurement noise covariance matrices as well as linearizing the dependency of the measurements of the Pipe Estimator and the Feature-Based Visual Odometry on the ones of the Complementary Filter.

5.2 Results

Complementary Filter

In order to estimate the accuracy of the Complementary Filter estimate, sample experiments are run while an April Tag was mounted on the head of the robot as shown in Figure 5.1. The tag motion estimate is used as ground truth, with its accuracy being empirically estimated to be of around 1 millimeter.

Figure 5.2 shows a sample tracking of the head position in the pipe cross-section plane, computed as ${}^{pipe}\begin{bmatrix}\sim & y & z\end{bmatrix}_{cam} = {}^{pipe}R_{CF} \cdot {}^{CF}t_{cam}$.

The root mean square error of position estimate has been of 1 centimeter. The rotation error is computed according to Lee et al [47]: $e_R = \frac{1}{2} \cdot tr(I - R \cdot R^T)$ and its RMSE is 0.006, with $I \in \mathbf{R}^{3\times3}$ identity matrix.

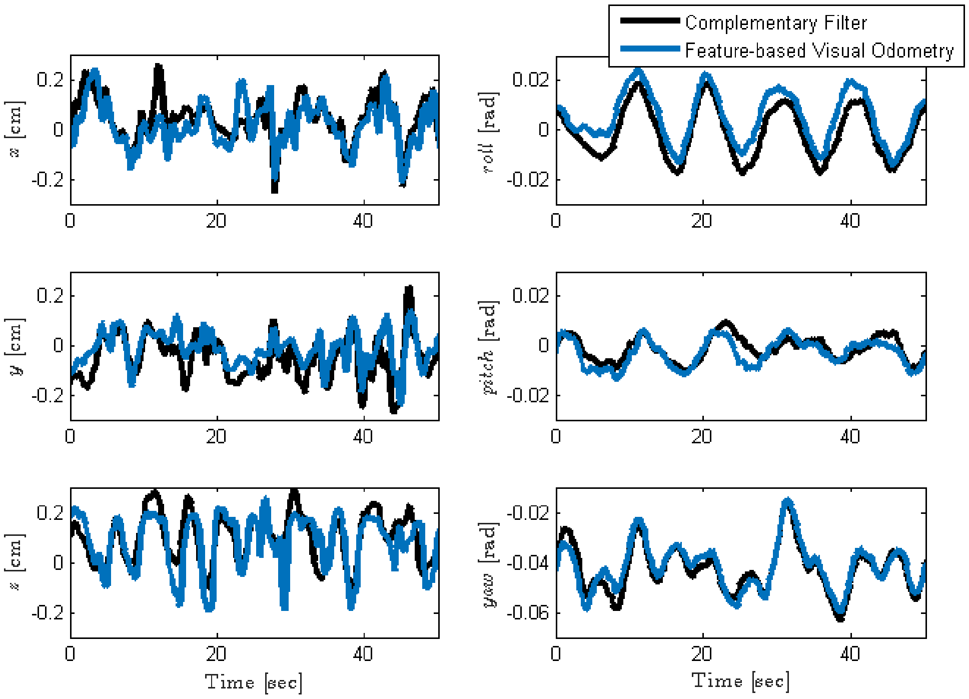


Figure 5.3: Differential local motion of the camera among two different frames. On the left the components of $^{cam'}t_{cam}$, on the right the Euler angles corresponding to $^{cam'}R_{cam}$. Black is Complementary Filter estimate, blue is Feature-based Visual Odometry estimate. The differential values are taken at 6Hz and are time filtered over 1 second to improve readability of the graph

Feature-based Visual Odometry

A sample comparison between the estimates of $^{cam'}T_{cam}$ by the Complementary Filter and by the Feature-based Visual Odometry is shown in Figure 5.3.

The differential value is computed every 170 milliseconds, i.e. every time a Feature-Based Visual Odometry estimate is available.

The Complementary Filter estimate is computed by finding the differential motion between two consecutive estimates, i.e. as $^{cam'}T_{CF'} \cdot {}^{CF}T_{cam}$.

Visual Joint Tracker

The accuracy of the translation estimate of the Visual Joint Tracker is compared to the estimate from manually annotated joint circles.

The accuracy of the latter depends on the reliability of the pinhole camera model. I tested it by comparing the estimate from the manually annotated circles to distance ground truth, measured from a meter. Its RMSE has been of 1 centimeter.

Once the pinhole camera model accuracy is determined, the circle tracker accuracy is evaluated. The estimate of the position of the joint in camera plane $^{cam}\begin{bmatrix} x & y \end{bmatrix}_{joint}$ has a RMSE of 3 mm, while the distance of the joint from the camera $^{cam}z_{joint}$ has a RMSE of 6 mm.

Figure 5.4 shows a sample comparison between joint coordinates according to manually annotated circles and according to the Visual Joint Tracker.

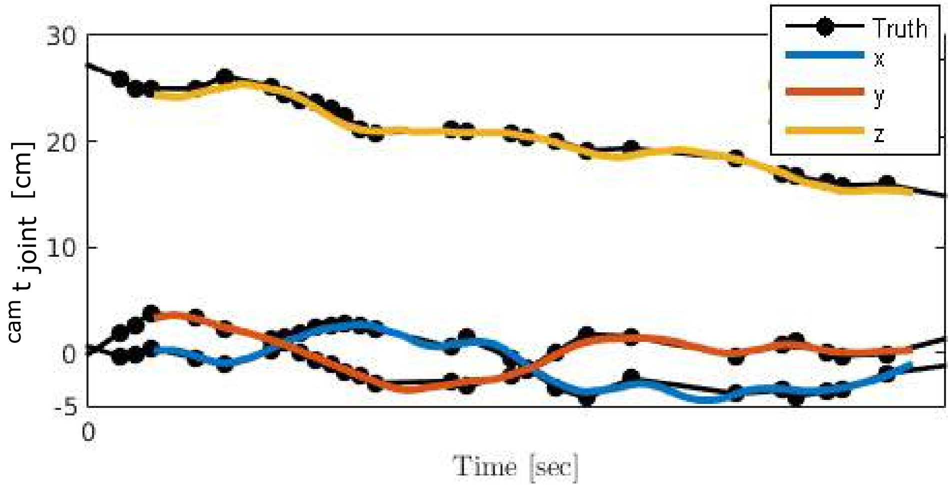


Figure 5.4: Position of the joint center in the camera frame. Black is ground truth, colored is the Visual Joint Tracker estimate of ${}^{cam}t_{joint}$. Blue, red and yellow are respectively x, y and z coordinates

Pipe Estimator

The Pipe Estimator has several roles in the framework and most of its estimate inter-twain with other block outputs. The only variable which the Pipe Estimator independently estimates is the angle of the bend the robot is going through, i.e. ${}^{pipe'}R_{pipe}$.

Figure 5.5 shows the trajectory over time of data related to the bend angle for different experiments.

In an ideal case the Complementary Filter estimate of the head yaw should follow the value of the real angle. The Pipe Estimator estimates the bend angles as the peak of the relative angle between the robot extremities.

In the performed experiments the Pipe Estimator has estimated the bend angle with an average error of 1%, while the Complementary Filter estimate had an average 10% mismatch.

Mapping and Odometry

Figure 5.6 show the top view of retrieved maps of the traversed pipes. The map is the representation of ${}^{world}T_{pipe}$ over time.

Figure 5.7 shows the robot trajectory over time for the same experiments of Section 5.2.

Ground truth is available only for the position of the robot, not for its orientation. Nevertheless, an approximate evaluation of the quality of the orientation estimate can be retrieved by the quality of the fitting of the body shape inside the pipe network. Sample frames are shown in Figure 5.8

The evolution over time of all the poses which are part of the EKF state is shown for a sample experiment in Figure 5.9.

A sample evolution over time of the uncertainty on the state is shown in Figure 5.10. The figure just shows the norm of the variance of the state components, e.g. the uncertainty of $x(1:4)(k) = {}^{world}q_{cam}$ is shown as the norm of $P(1:4,1:4)(k)$.

Figure 5.11 shows the evolution of the full covariance matrix trough a color-coded representation.

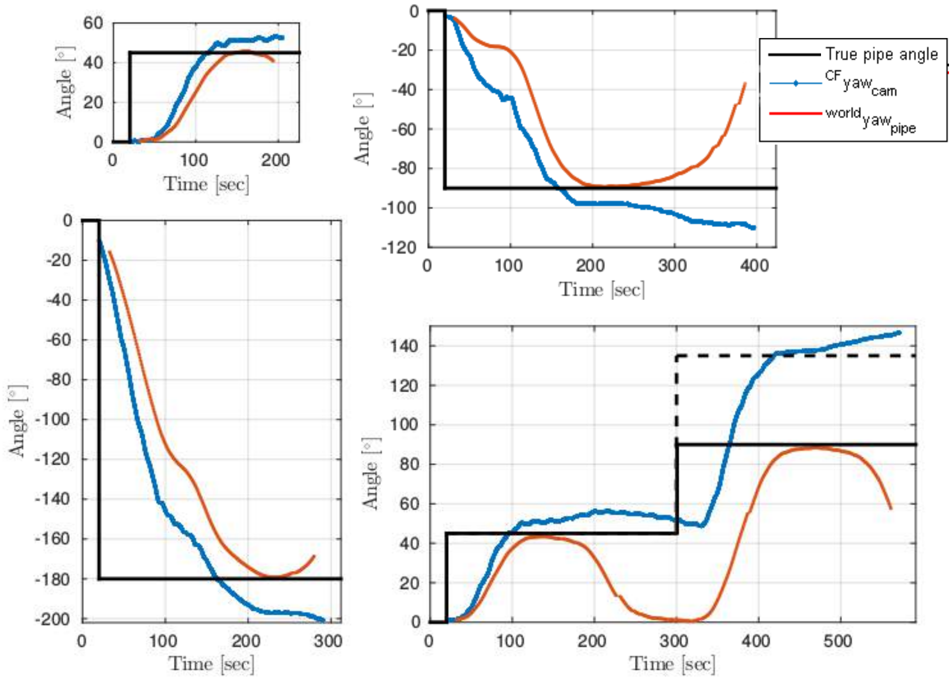


Figure 5.5: Angle of the bend that the robot is going trough. Black is ground truth, red is the angle between the robot extremities (its peak should be at the true angle value), and blue is the Complementary Filter estimate of head yaw (it should follow the true angle trajectory). In the bottom right figure the robot first goes through a 45° bend and then through a 90°, hence the reference head yaw of 45°+90° is shown in dotted black. The axes units are consistent among the plots

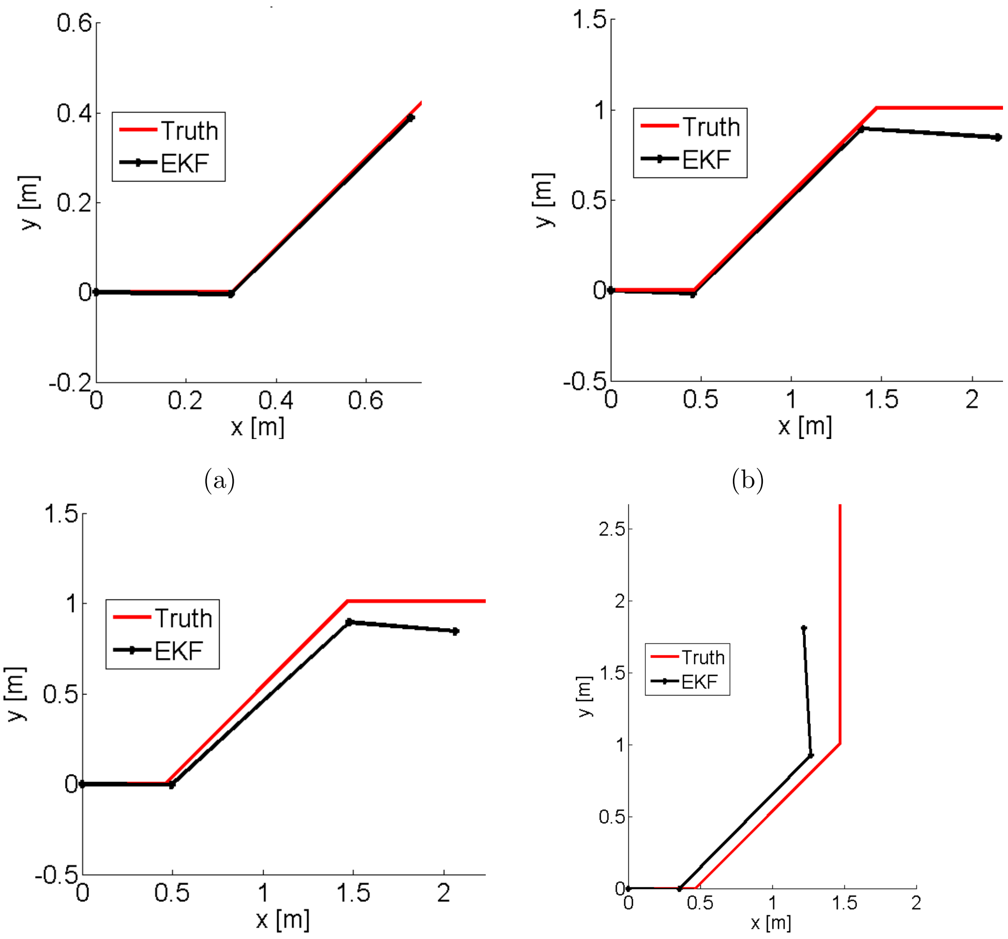
Figure 5.12 shows the comparison of the position odometry performed by the following estimators:

- EKF with only Complementary Filter and Feature Based Visual Odometry measurements. This corresponds to the traditional visual-inertial motion estimate
- EKF with Complementary Filter, Feature Based Visual Odometry and Visual Joint Tracker measurements
- EKF with Complementary Filter, Feature Based Visual Odometry and Visual Joint Tracker and Pipe Estimator measurements. This corresponds to the framework proposed in this work.

Figure 5.13 shows the periods of time in which the robot was estimated to be traversing a bend as well as when a pipe joint circle was visually tracked.

5.3 Discussion

While the amount of experimental tests does not allow for a statistically relevant analysis, several considerations about the strengths and the weaknesses of the framework can be drawn.



(a) (b)

(c) (d)

Figure 5.6: Top view of the map of the traversed pipe for four experiments. Black is the true map, red is the map retrieved from the Extended Kalman Filter. The axes are the world frame coordinates $^{world}x$ and $^{world}y$. The accumulated error for each of the shown map over total map length is: (a) 0.03 over 1.13 m (b) 0.19 over 4.64 m (c) 0.23 over 4.64 m (b) 0.68 over 4.73 m

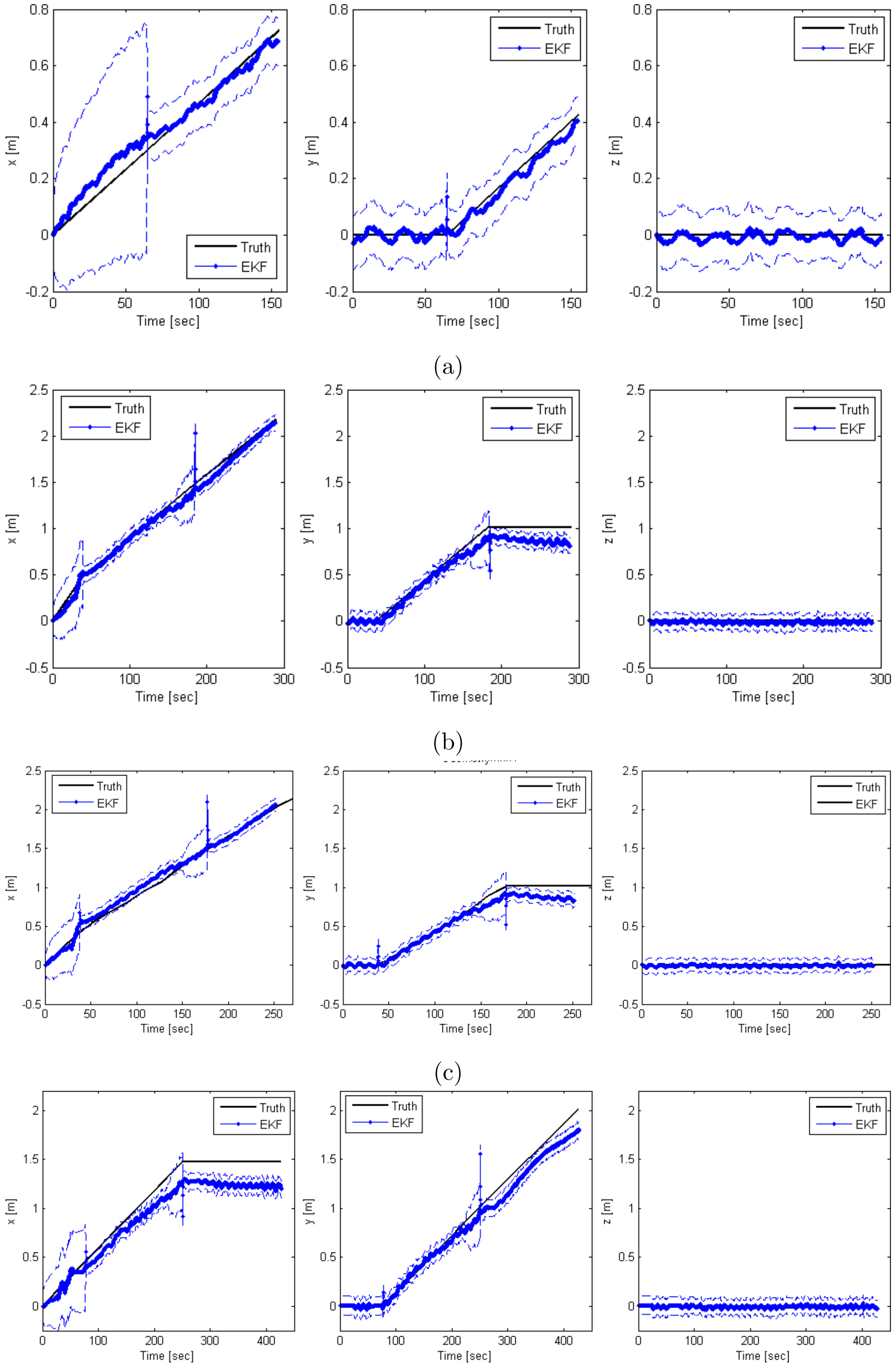
Individual estimates

As far as the accuracy of the estimates of the individual blocks of the framework is concerned, I was not interested in numerically computing the mismatch among the different estimates, since they strongly depend on the considered setup. Instead, I focused on checking that all of the estimators provides useful information into the sensor fusion filter. This has been shown true, as I briefly summarize in the following.

The Complementary Filter provides a tracking of the robot position in the pipe-cross-section which follows the April Tag ground truth.

The Feature-based Visual Odometry pose estimate is similar to the Complementary Filter one; this is an important finding, since the scarcity and poor quality of the visual features in the internal walls of the pipes posed serious challenges to the design and tuning of this block.

The Visual Joint Tracker shows to closely follow the translation ground truth, while its major drawback is track the joint for a shorter period of time with respect to



(a)

(b)

(c)

(d)

Figure 5.7: Estimates of the robot position in different experiments. Each of subfigures (a), (b), (c) and (d) shows the x, y and z components of ${}^{world}t_{cam}$ over time of a different experiment. Black is ground truth, blue is estimate from EKF and dashed blue is the three-standard-deviation range uncertainty of the estimate

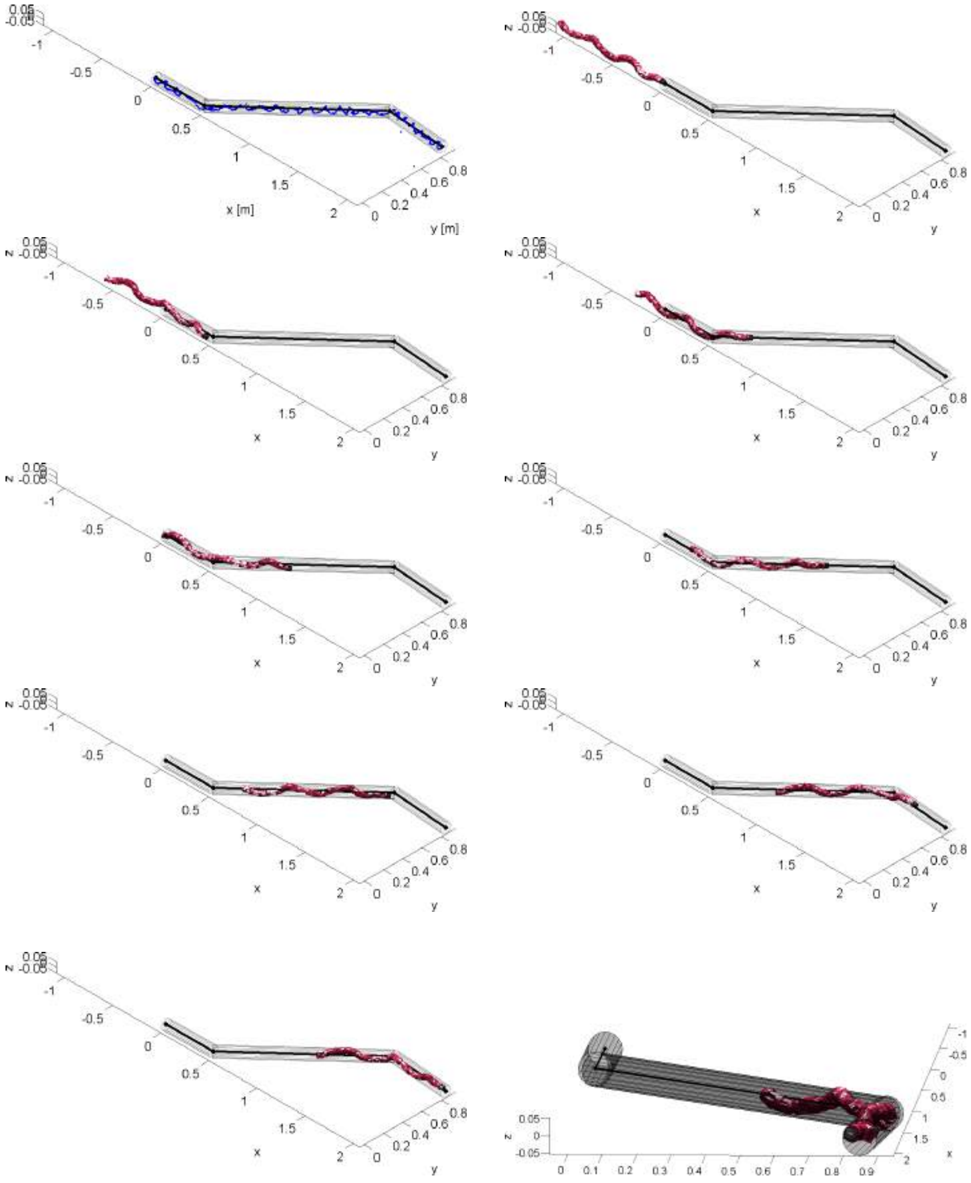


Figure 5.8: Graphic representation of the estimate of the pose of the robot ${}^{world}T_{cam}$ over time. On the top left corner, black is the reconstructed map, blue is the trace of the robot head position over time and the cylinders are the pipe segments. In the following subfigures, the graphic is graphically shown according to the position and orientation of the snake according to the current EKF estimate. Time flows left to right, then top to bottom. In the bottom right subfigure is a different perspective of the last robot pose

what a human annotator can do. This depends on the probability distributions of the false alarms and the target generated measurements, whose parameters have to be estimated. In practice, their choice is a trade-off between false positive (misclassified joint-generated circles) tracks and false negatives (misclassified noise-generated circles).
Finally, the Pipe Estimator shows to provide a more accurate estimate of the angles of the traversed pipe bends, allowing for a better reconstruction of the robot orientation with respect to inertial odometry alone. This outcome was to be expected because the major drawback of Complementary Filter orientation estimate is the drift in the yaw degree of freedom, due to the fact that it computed from the forward integration of the gyro readings. Hence, the exploitation of the kinematic constraints and the body shape compensate.

EKF

The mapping is a side product of the motion estimate, whose results are here analyzed.
The stronger take-home message is that the absolute measurements play the most important role in the performance of pose estimate. The absolute measurements are the ones from the Pipe Estimator, since it relies on the kinematics of the snake, and the Complementary Filter estimates of the head position in the pipe cross-section and the robot orientation in all degrees of freedom except yaw. On the contrary, the Feature-based Visual Odometry proved to be a weaker source of motion estimate since it just provides local differential odometry; this entails that it is prone to drift over time. Finally, the contribution of the Visual Joint Tracker was of little significance in the shown experiments; nevertheless, since the individual accuracy of its individual odometry showed to be promising, a better practical tuning of the circle Tracker is likely to improve the overall quality of the motion estimate.

Robot position

As far as the robot position is concerned, the quality of the estimate in the vertical direction is constant over time, since it is measured by the absolute estimate of the Complementary Filter. On the contrary, the quality of the estimate in the horizontal plane is noticeably greater when the robot is traversing a bend, as shown when comparing Figure 5.7 and Figure 5.13. This is a consequence of the availability of the estimate of the forward motion from the Pipe Estimator and has repercussions on the uncertainty of the estimate. In fact, the variance of the robot and Complementary Filter position blows up whenever the robot is not going through a bend, as shown in Figure 5.10. This is to be expected since the only available estimates of the forward motion come from differential measurements, Feature-based Visual Odometry and Visual Joint Tracker, whose errors accumulate over time. Figure 5.11 shows this aspect in the first three time instants (from time 0 to time 32 seconds) and in the instants at 159 and 175 seconds. In both cases, the variances and covariances of the position along the pipe main direction of the robot head and robot center of mass increase over time, till a bend is reached. In particular, in time instants 16 and 32 seconds, only the uncertainty of x-component of the translation increases since the pipe is aligned with world x-axis. This means that the variance of ${}^{world}x_{cam}$, of ${}^{world}x_{cam'}$ and of ${}^{pipe}x_{CF}$ as well as their co-variances with respect to each others increase. In time instants 159 and 175 seconds, the pipe is rotated of 45^o with respect to the world frame, hence also the uncertainty of ${}^{world}y_{cam}$, of ${}^{world}y_{cam'}$ blows up, while the co-variance between ${}^{pipe}x_{CF}$ and both of them also increases.

Robot orientation

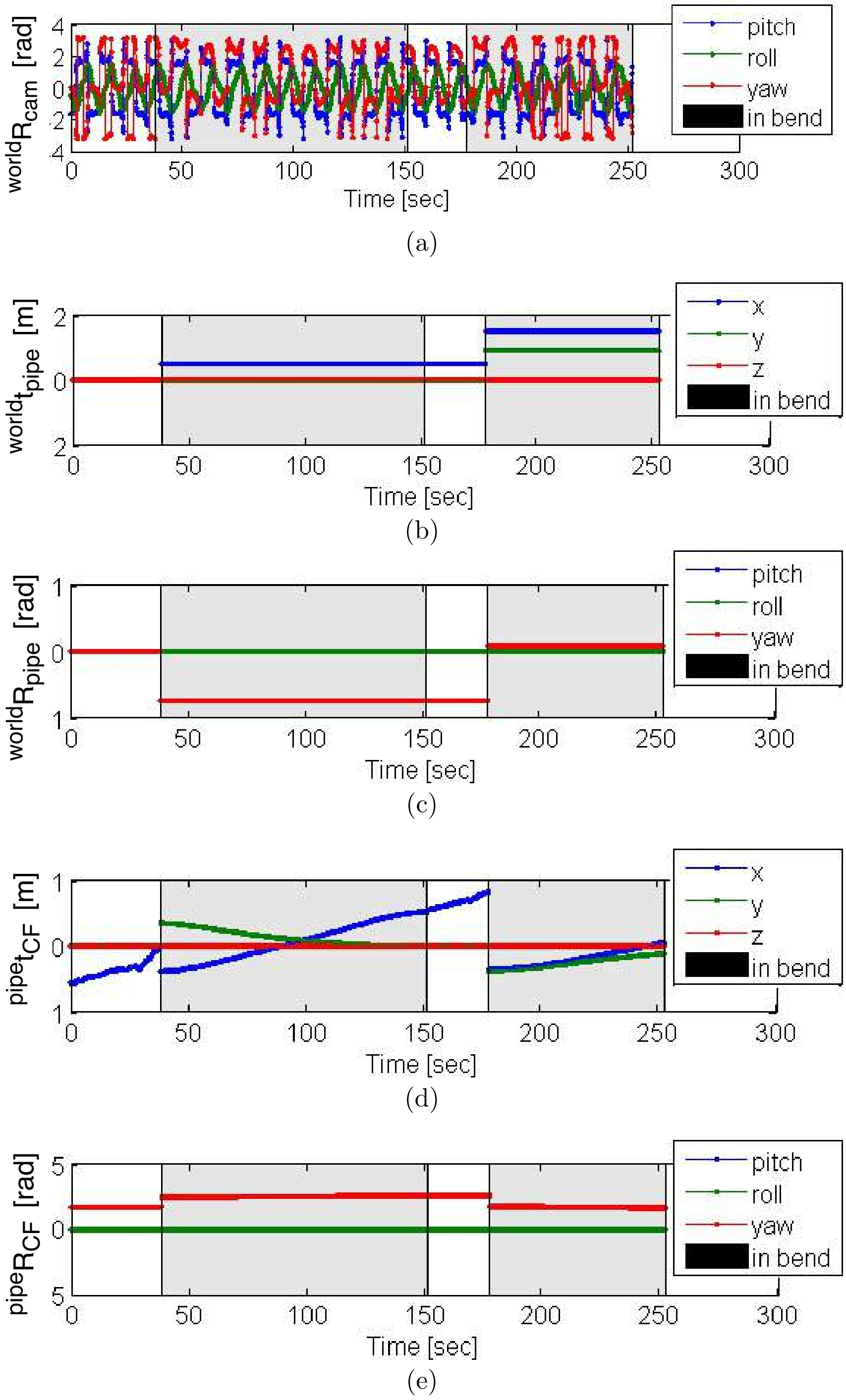
As far as the accuracy of the robot orientation estimate, no numerical results are available. Nevertheless, Figure5.8 shows that the robot body is mostly contained in the pipe segments at every point in time during the experiment. A full 6-DoF estimate must be accurate for this to happen in the time lapses of the experiment in which the robot traverses a bend. When the robot goes through a straight pipe, the piping map and the snake body can be consisted with each other up to two estimate mismatches: the rotation around the pipe main axis and the translation along it.

Other EKF state components

As far as the other components of the EKF state, two important aspects should be highlighted. First of all, Figure 5.9(d) supports the assumption that the robot center of mass position lies on the pipe main axis when the robot goes through a straight pipe segment. In fact, it is evident that ${}^{pipe}y_{CF}$ tends to zero when the robot finishes crossing a bend, i.e. from 100 to 150 seconds and from 225 seconds. Secondly, Figures 5.10 and 5.11 show that the position of the pipe frame is more and more uncertain the farther away in time is the robot switching from one pipe segment to the next. This follows from the fact that the only measurement of ${}^{world}t_{pipe}$ is available at these times, from the Pipe Estimator. In fact, the EKF state is detectable only at these points in time.

Performance comparison

Finally, the comparison of the full EKF with the visual-inertial EKF shows how the novel estimators of this work have improved the performance of the robot position estimate with respect to traditional solutions. In particular, the Pipe Estimator block has proven to give the utmost benefits to the framework. This is due to the previously exposed considerations, i.e. that it provides an absolute direct measurement of most of the filter state component. This is the reason why the uncertainty of Visual+Inertial estimates in Figure 5.12 blows up over time and their estimated trajectory so strongly mismatch from ground truth. As far as the Visual Joint Tracker contribution, it could be hardly appreciated in the presented experiments because of the limited time ranges in which a pipe joint has been visually tracked. Nevertheless, the performance of the Visual Joint Tracker per se is promising, implying that it can be a powerful once its circle tracker is accurately tuned.



(a)

(b)

(c)

(d)

(e)

Figure 5.9: Evolution over time of the EKF state for one experiment. For the sake of readability, rotations are shown in Euler angles. (a) Orientation of the camera in the world frame according to $^{world}q_{cam}$ (b) Position of the current pipe frame in the world frame $^{world}t_{pipe}$ (c) Orientation of the current pipe frame in the world frame according to $^{world}q_{pipe}$ (d) Position of the Complementary Filter frame in the current pipe frame $^{pipe}t_{CF}$ (e) Orientation of the Complementary Filter frame in the current pipe frame according to $^{pipe}q_{CF}$

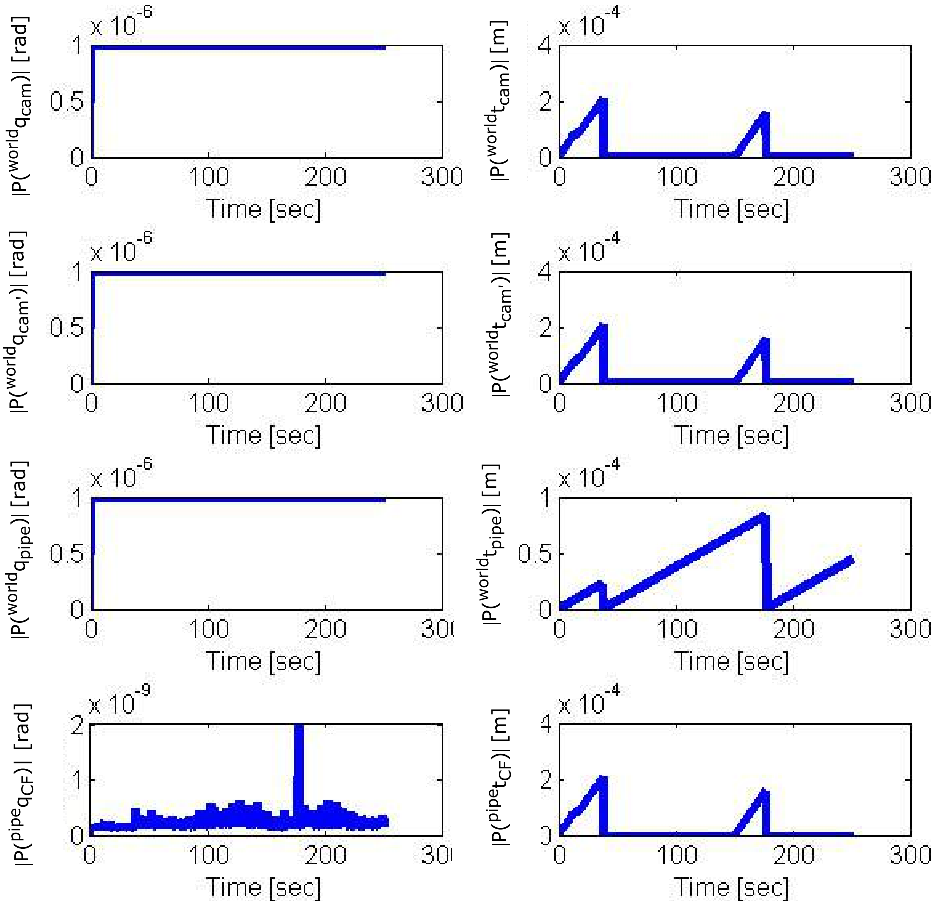


Figure 5.10: Evolution over time of the EKF state variance for one experiment. Each graph shows the norm of the variance matrix relative to a component of the state. In the order from left to right, then from top to bottom, the following variance matrix norms are shown: orientation of the robot head in the world frame ${}^{world}q_{cam}$, position of the robot head in the world frame ${}^{world}t_{cam}$, orientation of the robot head in the world frame in the previous filter iteration ${}^{world}q_{cam'}$, position of the robot head in the world frame in the previous filter iteration ${}^{world}t_{cam'}$, orientation of the current pipe frame in the world frame ${}^{world}q_{pipe}$, position of the current pipe frame in the world frame ${}^{world}t_{pipe}$, orientation of the Complementary Filter frame in the current pipe frame ${}^{pipe}q_{CF}$, position of the Complementary Filter frame in the current pipe frame ${}^{pipe}t_{CF}$

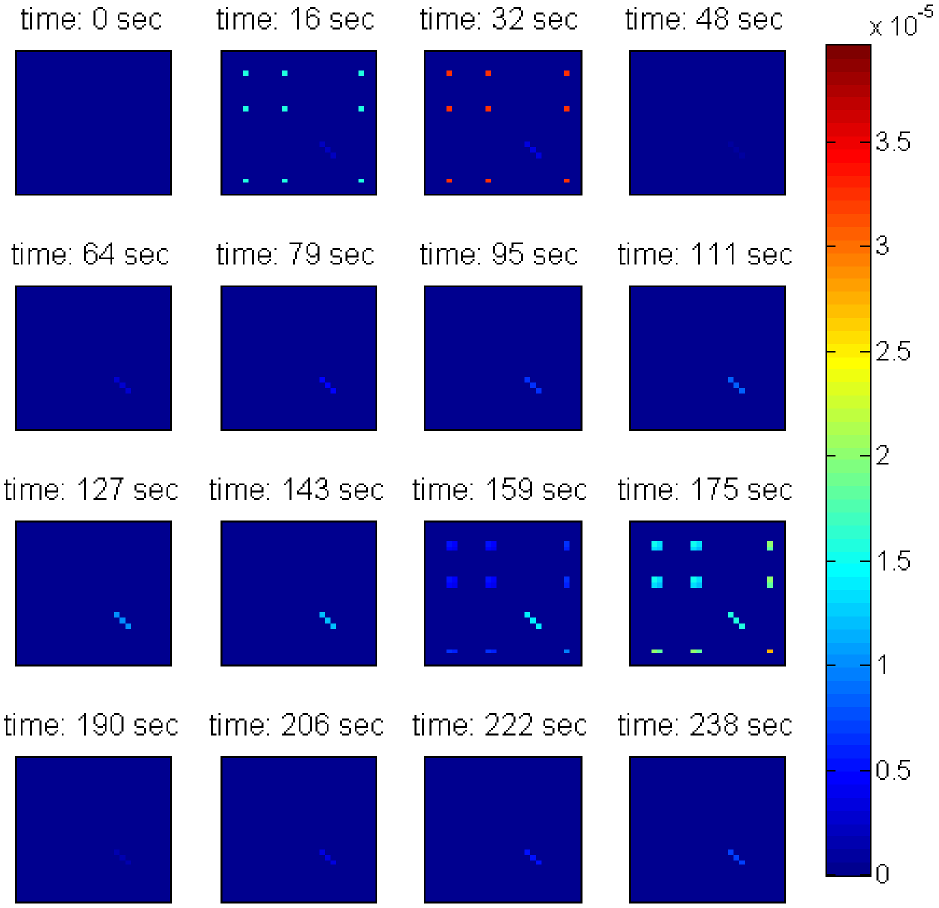
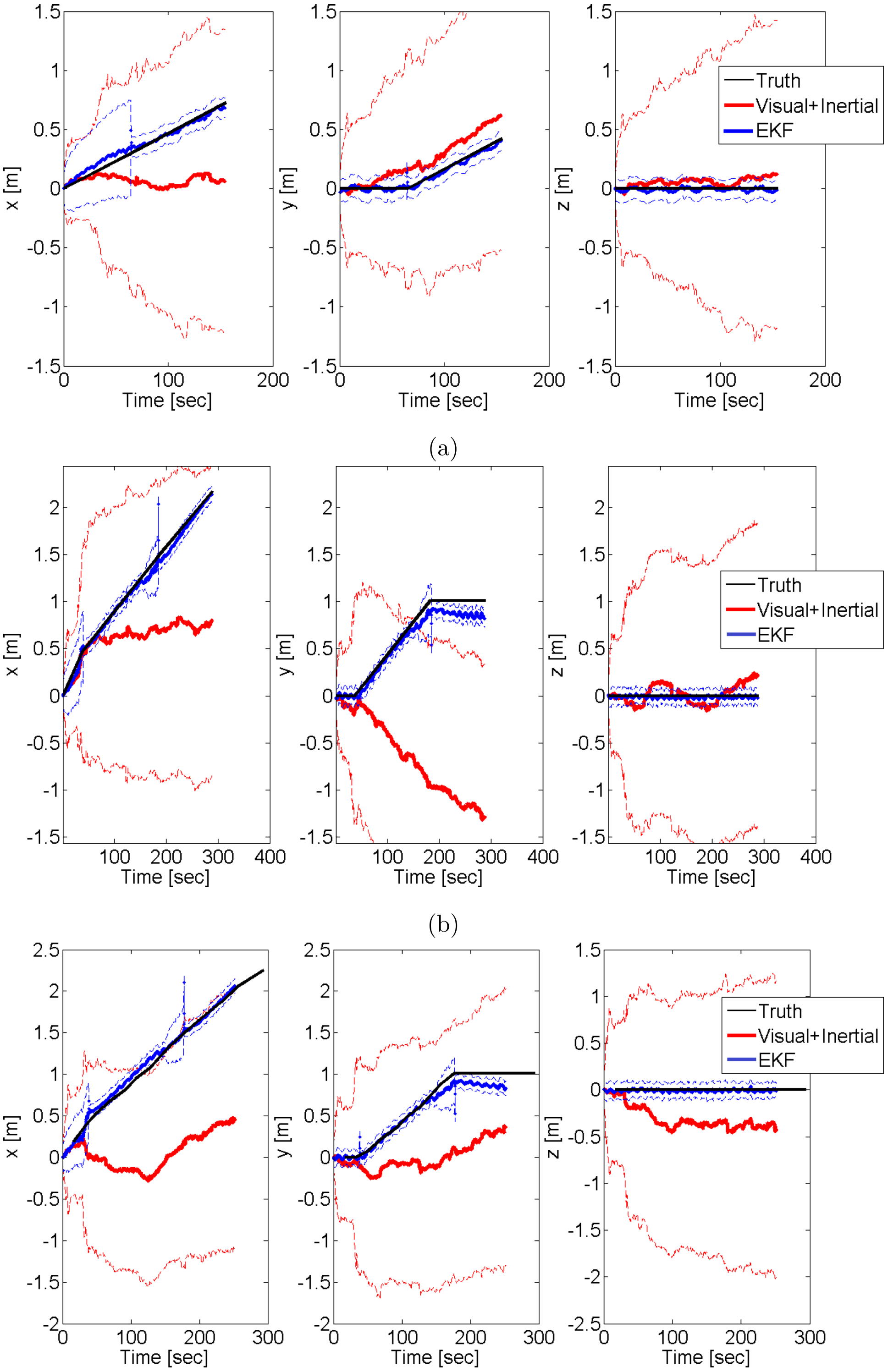


Figure 5.11: Evolution over time of the full covariance matrix of the state. Time flows from left to right, then from top to bottom. Each subfigure represents $P \in \mathbf{R}^{28\times28}$, the covariance matrix at an instant of time during the experiment. Its elements are color-coded according to the colorbar shown on the right



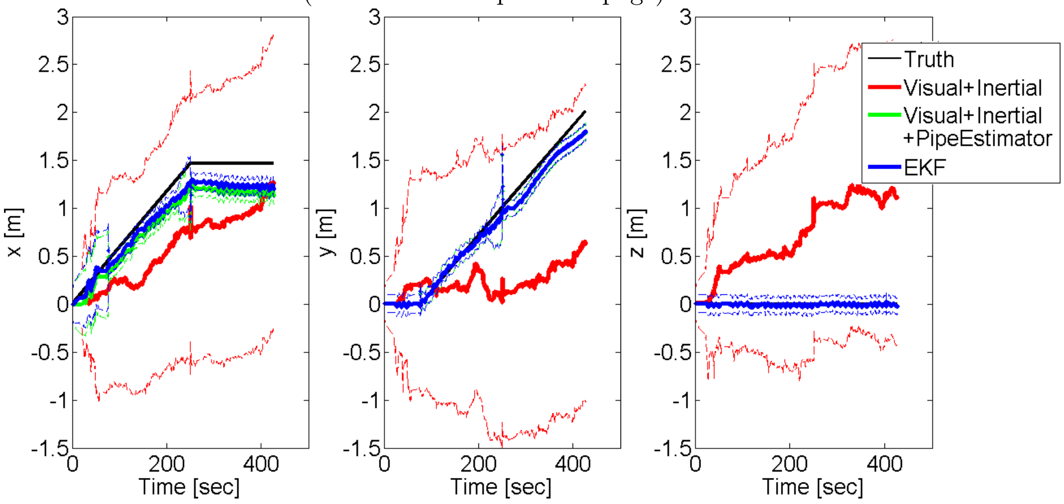
(a)

(b)

(c)

(continued in next page)

(continued from previous page)



(d)

Figure 5.12: Comparison among estimates of the robot position in different experiments. Each of subfigures (a), (b), (c) and (d) shows the x, y and z components of ${}^{world}t_{cam}$ over time of a different experiment. Black is the ground truth, blue is estimate from EKF with only Complementary Filter and Feature-based Visual Odometry measurements. Green is estimate from EKF with Complementary Filter, Feature-based Visual Odometry and Visual Joint Tracker measurements. In experiments (a), (b) and (c), green is not shown because covered by blue. Blue is from EKF with Complementary Filter, Feature-based Visual Odometry and Visual Joint Tracker and Pipe Estimator measurements

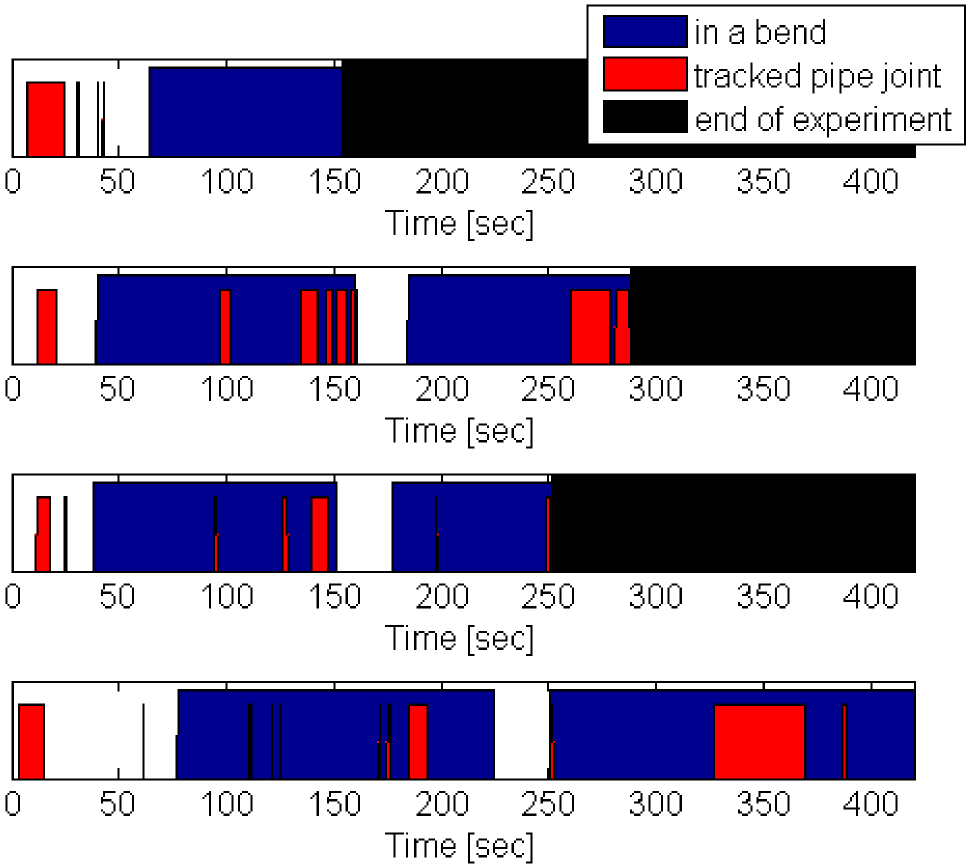


Figure 5.13: Availability of measurements over time. Each row corresponds to one of the experiments. On the x-axis is time. Red are the time instants in which a pipe joint is visually tracked, i.e. the measurement z_{VJT} is available. Blue are the time instants in which the robot is going through a bend, i.e. the measurement $z_{PE,t_{CF}}$ is available

Chapter 6

Conclusion

The aim of my work was to design and implement a solution for the motion estimation of snake-robot locomoting through a pipe network.

I wanted to overcome the limitations of the standard visual-inertial approach by taking advantage of the very structured environment. To this end, I have exploited two characteristics which are unique to the considered setup: the circles which are visible from the interior of the pipe when two pipe segments are connected together and the shape of the robot body. They both provide information about the robot motion: the position and size of the visual pipe-joint circles mirrors the translation of the robot and the robot shape allows us to infer the geometry of the traversed piping.

Hence, I devised two novel estimators and I formulated a sensor fusion framework which makes use of both traditional and original estimation techniques, so as to counterbalance the flows of each of them. The output of the framework is an estimate of the 6-DoF pose of the robot over time as well as a high-level map of the traversed path.

My idea proved to be successful in all the empirical results. First of all, the two novel estimators were able to convey meaningful estimates of the robot motion and the environment structure, which entitles them as being an important contribution to odometry frameworks in general. Secondly, my full framework solution showed improved performance in estimating the robot pose with respect to traditional visual-odometry solutions.

Another powerful characteristic of my solution is to be modular. This allows for an easy plug-and-play of the odometry components to adapt to different problem setups. This implies that several other kinds of robots would benefit from one or more elements of the proposed motion estimation framework. In particular, every robot which carries a frontal monocular camera for the inspection of the pipe can make use the Visual Joint Tracker and the Feature-based Visual Odometry. Moreover, robots whose shape adapts to the environment, regardless of their mechanics and locomotion methods, would profit of the proposed Pipe Estimator technique.

Future work directions have been explained in details in previous chapters. They can be summarized as: including the knowledge of the dependencies among the different individual motion estimates in the empirical implementation of the sensor fusion filter and including the current robot motion estimate in the motion model of the circular features visible at the pipe joints.

Acknowledgments

I am very grateful to Professor Choset and Professor Siegwart, as well as to Zeno Karl Schindler Foundation, for giving me the chance of doing this Thesis project. This experience at Carnegie Mellon University made me grow a lot, both at the professional and at the personal level.

I feel very lucky for having been exposed to the stimulating environment of BioRobotics Lab. It is so full of talented people as well as great guys at the human level, that I could not but being continuously motivated in doing my work at the best of my abilities. Many times I also received moral and technical support, from encouraging talks when I felt my Thesis project was failing miserably, to brain-storming sessions to organize my thoughts as well as day- and night-time support in running last-minute experiments. I am so thankful for all of it. A non-exhaustive list of thanks goes to: Professor Matt Travers, for being one of the best combinations of smartness and coolness which I ever met, Brad Saund and Ky Woodhard, for being always ready to listen to my technical problems and so good at giving good advises, Doctor Guillaume Sartoretti, for being an overwhelmingly good engineer and entertainer, Angela, for the never-ending availability and smiles, Doctor Glenn Wagner, for being so professionally competent and so painfully but helpfully honest in advising people, Doctor ChaoHui Gong, for being a role-model in humbleness, Doctor Alex Ansari, for providing smiling support whenever I needed it, and Puru Rastogi, for being a funny distraction.
Outside of the Lab, but still at CMU, I was lucky enough to meet Shushman Choundry, whose talent and dedication is only matched by his good heart.

Then, I would like to thank Francesco Ruscelli, for putting his admirable curiosity, availability and sense of humor in his roles of lab mate, house mate and friend.
I would also like to thank Haewon, Alessandro, Lavinia, Marianna, Arvind, Sylwia, Drew, Lair, Nirajan, Tejas, Angel, Alex, Gabriel, Florence, who, for the whole year of for just parts of it, made my free time enjoyable and memorable.
There are so many people who have been in my heart while in Pittsburgh. Thank you for never letting distance and time affect our friendship; to the Bologna friends Davide, Emilia, Maddalena, Lisa, Giulia, Andrea, Giacomo, Sofia, Barbara, Martina e Beatrice, and to the ETH-born friend gifts Gianluca, Matteo and Riccardo.

Finally, a huge thank you goes to my mother, whose generosity allowed me to do this Master course, as well as the in-between abroad internship experience and abroad Master Thesis. Thank you for making sacrifices for me, from handling the physical distance to taking planes and trains to cover it.

Last but not least, I am grateful to Lucas, for being a lovely companion, as well as an always-available support for my problems, both mathematical and practical. Your intellect and kindness made my year in Pittsburgh one which I hope to replicate soon, and over and over.

Bibliography

[1] W. Zhen, C. Gong, and H. Choset, “Modeling rolling gaits of a snake robot,” *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 2015-June, no. e, pp. 3741–3746, 2015.

[2] A. Kuwada, S. Wakimoto, K. Suzumori, and Y. Adomi, “Automatic pipe negotiation control for snake-like robot,” *IEEE/ASME International Conference on Advanced Intelligent Mechatronics, AIM*, no. 438, pp. 558–563, 2008.

[3] F. Enner, D. Rollinson, and H. Choset, “Motion estimation of snake robots in straight pipes,” *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 5168–5173, 2013.

[4] P. Corke, J. Lobo, J. Dias, and J. D. Peter Corke , Jorge Lobo, “An Introduction to inertial and visual sensing,” *The International Journal of Robotics Research*, vol. 26, no. 6, pp. 519–535, 2007.

[5] K. Konolige, M. Agrawal, and J. Solà, “Large-scale visual odometry for rough terrain,” *Springer Tracts in Advanced Robotics*, vol. 66, no. STAR, pp. 201–212, 2010.

[6] D. Nistér, O. Naroditsky, and J. Bergen, “Visual Odometry,” *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2004.

[7] G. Klein and D. Murray, “Parallel tracking and mapping for small AR workspaces,” *2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality, ISMAR*, 2007.

[8] F. Fraundorfer and D. Scaramuzza, “Visual odometry: Part I: The First 30 Years and Fundamentals,” *IEEE Robotics and Automation Magazine*, vol. 19, no. 2, pp. 78–90, 2012.

[9] D. Scaramuzza and F. Fraundorfer, “Visual Odometry Part II: Matching, Robustness, Optimization, and Applications,” *IEEE Robotics and Automation Magazine*, vol. 18, no. 4, pp. 80–92, 2011.

[10] J. Gui, D. Gu, S. Wang, and H. Hu, “A review of visual inertial odometry from filtering and optimisation perspectives,” *Advanced Robotics*, vol. 1864, no. October, pp. 1–13, 2015.

[11] S. Weiss and R. Siegwart, “Real-time metric state estimation for modular vision-inertial systems,” *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 231855, pp. 4531–4537, 2011.

[12] J. Kelly and G. S. Sukhatme, “Visual-inertial simultaneous localization, mapping and sensor-to-sensor self-calibration,” *Proceedings of IEEE International Symposium on Computational Intelligence in Robotics and Automation, CIRA*, pp. 360–368, 2009.

[13] J. Hol, T. Schön, H. Luinge, and P. Slycke, "Robust real-time tracking by fusing mea- surements from inertial and vision sensors," *Image Processing*, 2007.

[14] A. I. Mourikis and S. I. Roumeliotis, "A Multi-State Constraint Kalman Filter for Vision-aided Inertial Navigation," *IEEE Int. Conf. on Robotics and Automation (ICRA)*, pp. 3565–3572, 2007.

[15] P. Piniés, T. Lupton, S. Sukkarieh, and J. D. Tardos, "Inertial aiding of inverse depth SLAM using a monocular camera," *Proceedings - IEEE International Conference on Robotics and Automation*, no. April, pp. 2797–2802, 2007.

[16] T. Lupton and S. Sukkarieh, "Removing scale biases and ambiguity from 6DoF monocular SLAM using inertial," *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 3698–3703, 2008.

[17] S. Hilsenbeck, A. Moller, R. Huitl, G. Schroth, M. Kranz, and E. Steinbach, "Scale-preserving long-term visual odometry for indoor navigation," *2012 International Conference on Indoor Positioning and Indoor Navigation, IPIN 2012 - Conference Proceedings*, no. November, pp. 13–15, 2012.

[18] J. Ma, S. Susca, M. Bajracharya, L. Matthies, M. Malchano, and D. Wooden, "Robust multi-sensor, day/night 6-DOF pose estimation for a dynamic legged vehicle in GPS-denied environments," *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 619–626, 2012.

[19] J. S. Hu, C. Y. Tseng, M. Y. Chen, and K. C. Sun, "IMU-assisted monocular visual odometry including the human walking model for wearable applications," *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 2894–2899, 2013.

[20] M. Li and a. I. Mourikis, "High-precision, consistent EKF-based visual-inertial odometry," *The International Journal of Robotics Research*, vol. 32, no. 6, pp. 690–711, 2013.

[21] S. Lynen, M. W. Achtelik, S. Weiss, M. Chli, and R. Siegwart, "A robust and modular multi-sensor fusion approach applied to MAV navigation," *IEEE International Conference on Intelligent Robots and Systems*, pp. 3923–3929, 2013.

[22] M. Li and A. I. Mourikis, "Improving the accuracy of EKF-based visual-inertial odometry," *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 828–835, 2012.

[23] T. Lupton and S. Sukkarieh, "Efficient integration of inertial observations into visual SLAM without initialization," *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2009*, pp. 1547–1552, 2009.

[24] G. Nützi, S. Weiss, D. Scaramuzza, and R. Siegwart, "Fusion of IMU and vision for absolute scale estimation in monocular SLAM," *Journal of Intelligent and Robotic Systems: Theory and Applications*, vol. 61, no. 1-4, pp. 287–299, 2011.

[25] D. Strelow, "Motion Estimation from Image and Inertial Measurements," *The International Journal of Robotics Research*, vol. 23, no. 12, pp. 1157–1195, 2004.

[26] L. Kneip, S. Weiss, and R. Siegwart, "Deterministic initialization of metric state estimation filters for loosely-coupled monocular vision-inertial systems," *IEEE International Conference on Intelligent Robots and Systems*, pp. 2235–2241, 2011.

[27] N. Trawny and S. I. Roumeliotis, “Indirect Kalman filter for 3D Attitude Estimation,” *Technical Report Dept. of Computer Science & Engineering*, no. 2005-002, pp. 1–25, 2005.

[28] D. Rollinson, Y. Bilgen, B. Brown, F. Enner, S. Ford, C. Layton, J. Rembisz, M. Schwerin, A. Willig, P. Velagapudi, and H. Choset, “Design and architecture of a series elastic snake robot,” *IEEE International Conference on Intelligent Robots and Systems*, no. Iros, pp. 4630–4636, 2014.

[29] M. Tesch, K. Lipkin, I. Brown, R. Hatton, A. Peck, J. Rembisz, and H. Choset, “Parameterized and Scripted Gaits for Modular Snake Robots,” *Advanced Robotics*, vol. 23, pp. 1131–1158, 2009.

[30] D. Rollinson, A. Buchan, and H. Choset, “State estimation for snake robots,” *IEEE International Conference on Intelligent Robots and Systems*, pp. 1075–1080, 2011.

[31] W. T. Higgins, “A Comparison of Complementary and Kalman Filtering,” *IEEE Transactions on Aerospace and Electronic Systems*, vol. AES-11, no. 3, pp. 321–325, 1975.

[32] D. Rollinson and H. Choset, “Virtual chassis for snake robots,” *IEEE International Conference on Intelligent Robots and Systems*, pp. 221–226, 2011.

[33] N. Otsu, “A threshold selection method from gray-level histograms,” *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 9, no. 1, pp. 62–66, 1979.

[34] Jianbo Shi and C. Tomasi, “Good features to track,” *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition CVPR-94*, pp. 593–600, 1994.

[35] J.-y. Bouguet, V. Tarasenko, B. D. Lucas, and T. Kanade, “Pyramidal Implementation of the Lucas Kanade Feature Tracker Description of the algorithm,” *Imaging*, vol. 130, no. x, pp. 1–9, 1981.

[36] F. Fraundorfer, D. Scaramuzza, R. Siegwart, and M. Pollefeys, “Minimizing Image Reprojection Error For Relative Scale Estimation In Visual Odometry,” *Iros*, pp. 1–6, 2009.

[37] Wallis Engineering, “Sewer inspection mh e17c-057s to mh e17c-041s (23’ to end).vob,” https://www.youtube.com/watch?v=Y9ngUvuZpU&list=UU7Vl1uuMLZ5l5TtUtLSw-7w.

[38] J. Canny, “A computational approach to edge detection.” *IEEE transactions on pattern analysis and machine intelligence*, vol. 8, no. 6, pp. 679–698, 1986.

[39] H. Yuen, J. Princen, J. Illingworth, and J. Kittler, “Comparative study of Hough Transform methods for circle finding,” *Image and Vision Computing*, vol. 8, no. 1, pp. 71–77, 1990.

[40] Y. Bar-Shalom, F. Daum, and J. Huang, “The probabilistic data association filter,” *IEEE Control Systems Magazine*, vol. 29, no. 6, pp. 82–100, 2009.

[41] A. Rødningsby, “Multitarget Multisensor Tracking in the Presence of Wakes,” Thesis for the degree of philosophiae doctor, Norwegian University of Science and Technology, 2010.

[42] K. Buckl, “Sensor Fusion using the Kalman Filter,” http://campar.in.tum.de/Chair/KalmanFilter, 2005.

[43] G. Bradski, "OpenCV," *Dr. Dobb's Journal of Software Tools*, 2000.

[44] E. Morara, "Odometry and mapping inside pipes," https://github.com/biorobotics/Odometry_and_mapping_inside_pipes, 2016.

[45] "HERO Session," https://shop.gopro.com/cameras/hero-session/CHDHS-102.html, 2016.

[46] "HERO3+ Black Edition Field of View (FOV) Information," https://gopro.com/support/articles/hero3-field-of-view-fov-information, 2016.

[47] T. Lee, M. Leok, and N. H. Mcclamroch, "Geometric Tracking Control of a Quadrotor UAV on SE (3)," *49th IEEE Conference on Decision and Control*, no. 3, pp. 5420–5425, 2010.