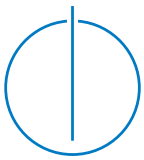
DEPARTMENT OF INFORMATICS

TECHNISCHE UNIVERSITÄT MÜNCHEN

Master's Thesis in Informatics

Photometric Odometry for Dynamic Objects

Anton Troynikov



DEPARTMENT OF INFORMATICS

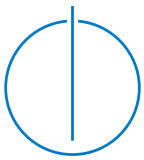
TECHNISCHE UNIVERSITÄT MÜNCHEN

Master's Thesis in Informatics

Photometric Odometry for Dynamic Objects

Photometrische Odometrie für Dynamische Objekte

Author:	Anton Troynikov
Supervisor:	Prof. Daniel Cremers
Advisor:	Nikolaus Demmel, M.Sc., Dr. Joerg Stueckler
Submission Date:	15. Apr. 2019



I confirm that this master's thesis in informatics is my own work and I have documented all sources and material used.

Munich, 15. Apr. 2019 Anton Troynikov

Acknowledgments

I would like to thank my advisors, Nikolaus Demmel and Dr. Jörg Stückler for their support, in particular through their detailed advice and in-depth technical discussions on all aspects of this work. I would also like to thank Prof. Dr. Daniel Cremers for the original inspiration for many of the ideas presented herein, as well as for the possibility of pursuing this thesis at the Chair for Computer Vision & Artificial Intelligence. Special thanks to Lucia Seitz for her intelligent discussion throughout this work's development, as well as her valuable comments in proof reading.

Abstract

In this master's thesis in Informatics, we present a direct photometric approach for using data from a visible light camera to estimate the motion of dynamic objects, as well as the motion of the camera itself. The presented algorithm differs from previous work in several key areas. First, through the use of dense direct photometric alignment, all available image data is used in a unified pipeline. Additionally, the presented algorithm differs from prior work in the use of keyframes, considerably reducing the preprocessing required for each image frame in the sequence. Finally, the presented approach does not attempt to track every object in the scene, but instead uses a conditional random field approach to perform binary segmentation using photometric information to identify, and subsequently track only moving objects. Through experiments with both real-world and synthetic data, we demonstrate the possibility of using direct photometric alignment to estimate the motion of dynamic objects, as well as camera egomotion, from sequences of image frames. In our evaluation we identify key limitations, and potential directions for future work.

Contents

1 Introduction

In order to interact safely and effectively with the real world, robotic systems such as autonomous drones, self-driving cars, and delivery robots must be able to accurately perceive the world in three dimensions, and in real-time. Recent advances in robotic perception have led to the increasing prevalence of these systems in the skies and on the streets around the world.

Figure 1.1: Moving objects identified in a street scene. Highlighted are cars, cyclists, and pedestrians, as well as other objects and regions of interest. [Cor+16]

Of particular importance to robotic perception is the ability to accurately detect moving objects, identify their type, and compute their path and velocity over time. For example, a self-driving car navigating a crowded intersection as in Figure 1.1 must be able to not only detect other road users, but also differentiate between parked and moving cars, cyclists, and pedestrians, and determine how they are moving.

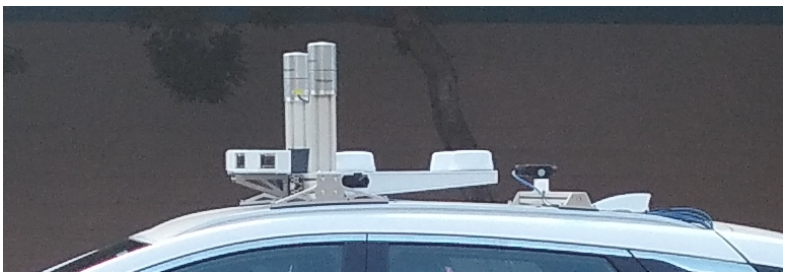


Figure 1.2: Typical example of a sensor array used in autonomous driving. Visible are long and short focal length cameras, LIDAR and laser scanners, as well as near and far-range radars. Author's photo.

Though the current state of the art in robotic perception for dynamic environments is reasonably effective in these tasks, it requires the use of a large array of many different sensors, including Light Detection and Ranging (LIDAR), Radar, Ultrasound, infrared 'structured light', and visible light cameras. A typical example of such a sensor array is shown in Figure 1.2. Such arrays are very expensive, heavy, and have high power consumption. Processing data from a wide variety of heterogeneous sensors is also computationally intensive and error prone. The size and weight of these arrays often makes them unsuitable for many applications including small drones and service robots.

Recently, significant advances have been made in perception algorithms that rely only on data from visible light cameras, particularly in the task of localizing the robot within the environment through which it travels - so called visual odometry. These algorithms have real-time performance and high accuracy. However, a key limitation is that these algorithms typically cannot deal with moving objects; these are either removed from the environment map, or else produce artifacts and errors.

1.1 Problem Statement

The objective of this thesis is to demonstrate the possibility of using only data from visible light cameras rigidly attached to a mobile robot to track the motions of dynamic objects as well as the motion of the robot itself.

Our approach is to extend a popular class of visual odometry algorithms, the so-called *direct* photometric algorithms, to the case of dynamic scenes, i.e. scenes containing

several moving objects other than the camera itself. The highly accurate estimates of the camera odometry that these algorithms have shown in static scenes make them a promising approach to estimating the odometry of dynamic objects.

Particular challenges include accurately identifying dynamic objects so they can be tracked, as well as operation in the presence of a wide variety of camera and object motions. We therefore also aim to experimentally identify the limits of photometric odometry when applied to the problem of tracking dynamic objects.

1.2 Outline

Chapter 2 provides an overview of existing approaches to visual odometry, as well as the tracking, detection and segmentation of dynamic objects. In particular, we highlight the challenges and limitations faced by these systems in the task of tracking dynamic objects from a moving camera. We outline the approach presented in this thesis, and contrast it with selected recent work in dynamic visual odometry.

Chapter 3 details the mathematical foundations of the approach developed in this thesis, covering rigid body transforms, direct photometric alignment, the use of Markov random fields for segmentation, and disambiguation through instance segmentation. The complete algorithm is then presented.

Chapter 4 details the evaluation of the algorithm developed in this thesis over representative data featuring dynamic objects. We demonstrate the effectiveness of the system in tracking dynamic objects, from both static and moving cameras. We also identify and discuss the experimentally observed limitations of the system.

Finally, in Chapter 5 we summarize the findings of the thesis, and present possible extensions and directions for future work.

2 Overview

In this chapter we review the existing literature in visual odometry and dynamic object tracking in computer vision, before giving a high-level overview of the approach proposed in this thesis. We subsequently highlight recent work in visual odometry and dynamic object tracking, and contrast it with the approach presented in this thesis.

2.1 Literature Review

The problem of estimating the motion of objects in the world, that is their odometry, can be cast as estimating some unknown state, comprising the object's position and orientation (pose) within it, given observations on the world. In the case of visual odometry, observations on the world arrive as a series of images, and the state to be estimated comprises the pose of the camera expressed as a rigid body transform $\boldsymbol{T}$, and the world geometry, typically as a set of points.

2.1.1 Visual Odometry

Most visual odometry algorithms are designed to estimate the motion of the camera through the scene. Many visual odometry algorithms form part of the pipeline for systems for simultaneous localization and mapping (SLAM), in which the 3D structure of the world is estimated alongside the motion of the camera within it, from a sequence of images.

While all visual odometry algorithms operate with image data, they differ in the way this data is processed. The two broad categories are keypoint-based, and direct approaches. Keypoint-based approaches appeared as the earliest implementations of visual odometry.

These approaches first process each image in the sequence to extract distinctive point features corresponding to landmarks in the world that can be reliably matched between images. These keypoints are designed to be robust to rotation and scale changes, as

well as illumination changes both due to scene lighting and camera auto-exposure. An early example of this approach is PTAM (paralell tracking and mapping) [KM09], while the state of the art monocular keypoint-based approach at time of writing is ORB-SLAM 2 [MT17].

An advantage of keypoint-based algorithms is the relatively simple representation, with each image resulting in a limited number of suitable keypoints to match. This ultimately reduces the required computation, resulting in improved throughput. Keypoints are also typically robust to orientation and lighting changes, as well as motion blur, making them robust in many real-world situations.

However, the relative sparsity of keypoints also means that much of the data available in the image is discarded. Additionally, the sparsity of keypoints mean that any given object in the scene may only have a few keypoints to match. This results in less accurate estimates of the object (or camera) motion, as well as the possibility to lose tracking as these few keypoints become occluded.

In order to overcome these and other limitations, in recent years so-called direct approaches have become increasingly popular. In contrast to keypoints, direct approaches do not use an intermediate representation for images, but rather use the color and intensity of pixels directly to perform tracking. This has the advantage that much more of the image data is used, resulting in robust tracking and accurate mapping. These are represented by dense approaches as in [KSC13] which take into account all visible pixels, as well as sparse approaches such as LSD-SLAM [ESC14] where only a selected subset of the pixels are used.

Due to the need to process a large number of data elements, real-time performance for direct approaches was often difficult to obtain, resulting in the need for asynchronous optimization of the estimated odometry, or trading increased speed for degraded accuracy. Additionally, direct approaches have been shown to be more vulnerable to illumination changes and other image artifacts due to e.g. lens and rolling shutter distortions than keypoint based approaches.

More recently, direct formulations capable of real-time performance, and robust to illumination changes and image artifacts, have proven successful. In particular, DSO [EKC18] demonstrated real-time performance coupled with a high degree of robustness through a principled choice of sparse image points, which are processed in a direct

fashion.

2.1.2 Dynamic Objects in Computer Vision

Dealing with dynamic objects, i.e. those objects moving relative to the camera, is a classic problem in computer vision with a vast associated literature. Of particular interest are the problems of detection, tracking, and reconstruction of dynamic objects. Algorithms that solve these problems are often interrelated, for example tracking through detection, or tracking for reconstruction.

Dynamic objects may be either rigid bodies, such as cars or trucks, or else undergo non-rigid deformations, for example pedestrians or cyclists. In practice, deformable objects are often approximated as rigid for the purposes of tracking.

Detection and Segmentation

The detection problem amounts to identifying an object of interest in an image. It is closely related to the problem of segmentation, which deals with determining which pixels in an image belong to the object of interest. Classically, this problem has been solved through keypoint or feature matching, using a similar approach to feature extraction as in the keypoint formulation of visual odometry [V+01][Ozu+10][NP14]. Known keypoints or features on the object of interest are associated with keypoints in an image, resulting in an approximate bounding box detection.

Other approaches have sought to exploit additional cues, such as the overall colour distribution in the object of interest, to assign a probability to each image pixel based on the likelihood of belonging to the object, using e.g. expectation maximization in a Gaussian mixture model [Li+04][SS05]. Thresholding the pixels based on probability results in a segmentation of pixels belonging to the object, which can be further refined through the use of energy minimization approaches such as Markov random fields [ZBS01].

More recently, Machine Learning approaches have overtaken classical approaches to detection and segmentation, in terms of both precision and accuracy. These approaches rely on a large database of labeled object images to train artificial neural networks in the detection task. A popular approach is to slide a bounding box over the image, and apply a learned classifier at each step. The bounding boxes giving the highest

likelihood for the desired class are then assigned as the predicted bounding box of a detected object [Red+16][Liu+16].

A key advantage of machine learning approaches to tracking and segmentation is their ability to generalize over a class of objects; for example, to detect all cats in an image, even if the specific animals were not present in the training data. However, machine learning approaches typically struggle with the problem of detecting a specific object over many images.

In addition to requiring a large amount of training data, machine learning approaches are also typically rather computationally intensive, both during training and during detection / segmentation. Additionally, the trade-offs are not yet sufficiently well understood; recent research has shown that it is possible to create adversarial images that fool learned detectors, mis-classifying as stop sign as a speed sign, with small changes to the input not visible to humans [Goo+14][KGB16].

Tracking

The tracking problem can be expressed as determining the motion of an object of interest relative to the camera, from a sequence images. Estimating the motion of objects typically requires additional information about the 3D structure of the object being tracked, whether from a depth sensor or a previously constructed 3D template, in order to correctly estimate the motion of the object through space.

The simplest approaches to object tracking rely on the idea of tracking-by-detection, that is simply identifying the object of interest by searching for it in each image individually, and then associating these detections between subsequent images in order to reconstruct the motion of the object [Bre+09][ARS10].

More sophisticated approaches employ motion models and filtering in order to predict the position of the object in a subsequent frame and to initialize the detection process in a particular region, rather than searching over the entire image. The difference between the predicted and detected positions are the subsequently fused for the next prediction, and to update the motion model.

Tracking can also be accomplished through object templates; representations of the 3D structure, color and texture of objects of interest. These templates can then be

matched across image sequences in a similar way to keypoint matching, but taking into account much more information about the object of interest [ZJD00][ZLY12]. Template tracking can also incorporate a motion model, in the same way as keypoint matching. Such approaches, as in as in Tan et. al. [TI14] often make use of data from a depth sensor, in order to improve precision and accuracy.

2.2 Proposed Approach

The direct photometric alignment approach to the visual odometry problem estimates rigid body motions from images by estimating the rigid body transform that best photometrically aligns the pixel intensities in an image frame, with those in a keyframe that includes both intensity and depth information.

In traditional examples of this approach such as [ESC14][EKC18][Sco+18], the estimated transform is interpreted as being due to the motion of a camera through an assumed static world. Where points in the world produce photometric inconsistencies, they are typically down-weighted or removed from consideration for estimating the transform.

This means that in these approaches, objects which are also moving relative to the world are not modeled and actively excluded. They cannot be tracked, and may degrade the quality of the estimated camera motion.

2.2.1 Key Ideas

The algorithm detailed in this thesis seeks to overcome this limitation through extending the direct photometric alignment approach to also deal with dynamic objects. Rather than discarding or down-weighting points that violate the static-world assumption, we instead seek to segment sets of points corresponding to moving rigid bodies, and subsequently estimate rigid body transforms for each of them.

In order to do so we note two important observations. First, if an estimated transform does not correspond to the motion of a given object, points on the object will result in photometric inconsistencies. Thus we may use photometric inconsistencies which remain after the estimation of a transform through direct alignment as cues to detect and segment independently moving objects.

Second, direct alignment may be used to estimate not only the motion of a camera relative to a static world, but also the motion of a camera relative to any rigid object. Thus, if it is possible to segment independently moving objects, we may perform direct

alignment in order to estimate a rigid body transform between the object and the camera.

Our algorithm combines these observations in a framework that jointly detects, segments, and estimates the rigid body motion of independently moving objects, as well the motion of the camera relative to the world, on the basis of image data alone.

In applying the direct photometric approach to the problem of tracking dynamic objects, we preserve the advantages of using all available image information, leading to improved accuracy and robustness. A unified approach allows for a simpler tracking pipeline, since only image data is used in every step, rather than being transformed between domains. Additionally, placing the object tracking problem on the same mathematical foundation as other direct photometric odometry algorithms allows for future extension using approaches developed in those domains.

2.2.2 Algorithm Outline

Though several variant approaches to the problem of dynamic visual odometry have been proposed, that which is explored in this thesis is as follows.

Given a keyframe, consisting of an image with pixel intensities, and associated depths, the algorithm takes as input observations in the form of images with pixel intensities. The algorithm seeks to explain the observed input in terms of rigid body transforms applied to sets of points corresponding to objects in the world, including independently moving objects as well as the static background.

We perform direct alignment to arrive at an initial rigid body transform estimate. We then determine which points produce photometric inconsistencies, given the estimated transform. We segment these to an outlier set, and re-estimate the transform without these points until the outlier set no longer grows or the estimated transform no longer changes.

Subsequently, we associate points in the outlier set to independently moving objects, using instance segmentation. For each object instance found in this way, we repeat the direct alignment process and the outlier segmentation to arrive at an estimate for rigid body transforms which best explain each part of the segmented input, arriving at a set

of consistent motions and object segments.

We discuss each step in detail in the subsequent chapters.

2.3 Related Work

The problem of estimating the camera motion as well as the motion of dynamic objects relative to the world has recently seen increased interest, as robots are increasingly tasked with operating in highly dynamic environments such as public roads, warehouses, hospitals, and many others. Much of the research into this problem is relatively recent, due to the overall complexity of the problem, higher computational requirements for real-time performance, and the lack of suitable datasets that include ground-truth information for the structure and motion of objects, the world, and the camera.

Recent work as in [Bar+17] has shown that the performance of visual camera odometry algorithms significantly degrades in highly dynamic environments. The presence of many moving objects often results in significantly decreased camera localization accuracy.

In order to overcomes this limitation, some approaches exclude dynamic objects in the scene, by segmenting it into foreground and background. This is accomplished through either explicit object detection as in [Bes+18] which leverages semantic and geometry segmentation, implicitly through geometric clustering over depth images as in [Sco+18], or clustering from optical flow as in [Che+18]. While these approaches show an improvement in estimating the camera egomotion estimate in dynamic scenes, they do not track the dynamic objects themselves.

In contrast, the approach proposed in this thesis estimates the motion of dynamic objects, while refining the camera egomotion estimate.

Several approaches for the simultaneous estimation of both dynamic object motions, and camera egomotion, have been proposed.

In [Xu+18], dense semantic segmentation is performed for each frame using a neural network to detect visible objects. Motion residuals in image and geometry are then computed for each object to determine whether it is moving, and subsequently moving objects are tracked. The motion estimate for the camera is first computed from all

colour and depth information in each frame, and subsequently refined by excluding moving objects.

In [Ose+17], a large number region proposals are used to segment input images into class-agnostic regions. These are then associated with 3D geometry information, computed from stereo matching. Proposals are then aggregated into objects, and their motions estimated, through the selection of temporally consistent motion and region hypotheses. Tracking is performed only relative to the camera, camera egomotion is not estimated.

Our proposed approach differs from these in several important ways. Rather than detecting every visible object, or object segment, and then computing motion estimates for each, we instead first determine which regions of each frame are inconsistent with the hypothesis of a static scene, and subsequently determine which of these inconsistencies can be associated with objects. This significantly reduces the amount of required computation. We subsequently track both object and camera egomotion.

Tracking dynamic objects and camera ego motion without the use of segmentation is demonstrated in [JGN18]. This work leverages keypoint matching between pairs of stereo images, as well as temporally over the sequence of frames. A tracklet for each keypoint is then computed in the camera coordinate frame. Tracklets are then grouped to object labels through the use of rigid body motion, and geometric neighborhood assumptions. The camera egomotion is then estimated as the motion relative to the label containing the greatest number of keypoints.

The main distinction between [JGN18] and our approach is our use of direct photometric alignment to leverage all image information associated with an object, rather than only sparse keypoints. This gives the same advantages as direct photometric odometry over keypoint based odometry, in the object tracking domain.

One final distinction between our approach and those discussed in this section is our use of keyframes. In each of the works discussed herein, every frame must be fully processed before use, whether via semantic segmentation, geometric clustering, region segmentation, or stereo matching. In contrast, our approach only requires additional processing for those frames promoted to keyframes. This again reduces the total amount of computation required, as well as opening the door for future improvements, for example via sliding window bundle adjustment as in [EKC18].

We now proceed to detail the mathematical foundations of the proposed approach.

3 The Dynamic Photometric Odometry Algorithm

In this chapter we detail the mathematical foundations of the dynamic photometric odometry algorithm. We detail rigid body motions, including relative motions and parametrizations. We then derive the direct photometric alignment algorithm for transform estimation. Next we detail binary image segmentation through the use of conditional random fields, and the use of instance segmentation to associate photometric outliers with objects. Finally, we present the complete algorithm.

3.1 Rigid Body Motion

An object moving through space relative to some coordinate frame, while preserving its size and shape, is said to undergo a rigid body transform. Formally, a rigid body transform T in 3-dimensional Euclidean space is a map;

$$T : \mathbb{R}^3 \mapsto \mathbb{R}^3 \tag{3.1}$$

such that for any points $\boldsymbol{p} \in \mathbb{R}^3, \boldsymbol{q} \in \mathbb{R}^3$ belonging to the set of points $\boldsymbol{X}$ on an object undergoing a rigid body transform;

$$\|\boldsymbol{p} - \boldsymbol{q}\| = \|T(\boldsymbol{p}) - T(\boldsymbol{q})\| \quad \forall \boldsymbol{p}, \boldsymbol{q} \in \boldsymbol{X} \tag{3.2}$$

$$T(\boldsymbol{p}) \times T(\boldsymbol{q}) = T(\boldsymbol{p} \times \boldsymbol{q}) \quad \forall \boldsymbol{p}, \boldsymbol{q} \in \boldsymbol{X} \tag{3.3}$$

Such rigid body transforms consist of a rotational component and a translational component. Various representations of the rotational component exist, including quaternions and axis-angle representations. We employ a 3×3 orthogonal rotation matrix $\boldsymbol{R}$, which is a matrix in the so-called special orthogonal group $SO(3)$. The translation component may be expressed as a vector $\boldsymbol{t} \in \mathbb{R}^3$. A rigid body transform T may then be expressed as;

$$T(\boldsymbol{p}) = \boldsymbol{R}\boldsymbol{p} + \boldsymbol{t} \tag{3.4}$$

By expanding the representation of the point $\boldsymbol{p}$ to so-called homogeneous coordinates, $\bar{\boldsymbol{p}} = (x, y, z, 1)^\top = \left[\boldsymbol{p}^\top, 1\right]^\top$, we may express the transform T as a 4×4 matrix $\boldsymbol{T}$ consisting of the rotation matrix $\boldsymbol{R} \in SO(3)$ and the translation vector $\boldsymbol{t} \in \mathbb{R}^3$;

$$\boldsymbol{T} = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \\ \boldsymbol{0} & 1 \end{bmatrix} \tag{3.5}$$

The resulting matrix $\boldsymbol{T}$ is in the special Euclidean group, $SE(3)$. We may apply the transform to the point in homogeneous coordinates as;

$$T(\boldsymbol{p}) = \boldsymbol{M}\boldsymbol{T}\bar{\boldsymbol{p}} \tag{3.6}$$

where $\boldsymbol{M}$ is the 3x4 matrix which converts from homogeneous to euclidian coordinates, given as $[\boldsymbol{I}|\boldsymbol{0}]$ where $\boldsymbol{I}$ is the 3x3 identity matrix.

This matrix expression of T is particularly useful as it allows us to compute the inverse transform as the matrix inverse;

$$\boldsymbol{T}^{-1} = \begin{bmatrix} \boldsymbol{R}^\top & -\boldsymbol{R}^\top \boldsymbol{t} \\ \boldsymbol{0} & 1 \end{bmatrix} \tag{3.7}$$

as well as chaining transforms as matrix products;

$$T_2(T_1(\boldsymbol{p})) = \boldsymbol{M}\boldsymbol{T}_2\boldsymbol{T}_1\bar{\boldsymbol{p}} \tag{3.8}$$

Conveniently, the identity transform is represented as the 4×4 identity matrix, $\boldsymbol{I}$.

3.1.1 Parametrizing Rigid Body Transforms: Lie Algebra

A key aspect of applying this definition of rigid body transforms in optimization problems, as we shall do subsequently, is to choose an appropriate parametrization. The translation component of a rigid body transform in 3 dimensions, $\boldsymbol{t}$ has three elements and three degrees of freedom. However the rotation component, when represented as a 3×3 matrix, has nine elements but only three degrees of freedom; this is due to the requirement that the matrix is orthogonal, and that the corresponding row and column vectors are unit vectors.

Representing these constraints directly is cumbersome, and limits the applicable optimization strategies. However, it is the case that the group $SE(3)$ is also a differentiable manifold, and hence is a so-called Lie group. Hence, the group $SE(3)$ has an associated Lie algebra, associated with the tangent space at the identity transform I. Elements in the the Lie algebra are expressed as $\hat{\boldsymbol{\xi}} \in \mathfrak{se}(3)$.

The Lie algebra representation of $\boldsymbol{T}$, $\hat{\boldsymbol{\xi}}$ can be expressed as a vector of six parameters, $\left[\boldsymbol{\omega}^\top, \boldsymbol{v}^\top\right]^\top \in \mathbb{R}^6$. Here $\boldsymbol{\omega} \in \mathbb{R}^3$ determines the rotation, and is analogous to angular velocity, while $\boldsymbol{v} \in \mathbb{R}^3$ determines the translation, and is analogous to the linear velocity.

Conversion from $\hat{\boldsymbol{\xi}}$ to $\boldsymbol{T}$ is accomplished via the exponential map;

$$\exp : \mathfrak{se}(3) \mapsto SE(3) \tag{3.9}$$

Which has a closed-form expression via matrix exponentiation and the Rodriguez formula, and from $\boldsymbol{T}$ to $\hat{\boldsymbol{\xi}}$ via the logarithm map;

$$\log : SE(3) \mapsto \mathfrak{se}(3) \tag{3.10}$$

Note that $\hat{\boldsymbol{\xi}} = \mathbf{0}$ corresponds to the identity transform.

3.1.2 Relative Rigid Body Motion

In the standard formulation of most visual odometry algorithms, all observed changes are attributed to the rigid body motion of the camera through space, relative to a fixed background or world frame. In the case of a dynamic scene however, we must also account for the rigid body motion of other objects relative to the camera, as well as the motion of those objects relative to the fixed world frame.

Transforms and Poses

The rigid body transform $\mathbf{T}_{AB}$, when applied to a point in coordinate frame B, transforms it to the coordinate frame A. That is, for a point $\boldsymbol{p}$ seen at $\bar{\boldsymbol{p}}_B$ in homogeneous coordinates from frame B, the same point is seen at $\bar{\boldsymbol{p}}_A = \mathbf{T}_{AB}\bar{\boldsymbol{p}}_B$ in frame A.

We note that $\mathbf{T}_{AB}$ also corresponds to the relative position and orientation of the coordinate frame B as seen in coordinate frame A; this is called the *pose* of B in the

coordinates of A. Thus $\mathbf{T}_{AB}$ defines both a pose and a coordinate transform between two frames. These relationships are represented in Figure 3.1.

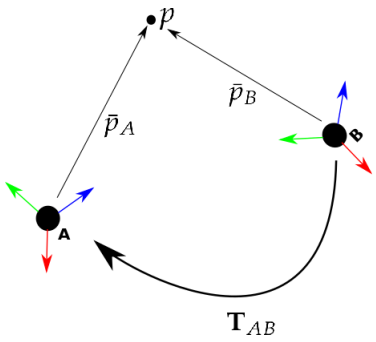


Figure 3.1: Coordinate reference frames. $\mathbf{T}_{AB}$ represents both the relative pose of B as seen from A, as well as the rigid body transform of $\bar{\boldsymbol{p}}_B$ to $\bar{\boldsymbol{p}}_A$

These simple relationships are sufficient to describe static rigid bodies, or the position of a moving rigid body relative to a fixed frame. However, they are insufficient for the case where we wish to describe the rigid body motions of multiple moving objects.

Transforms over time

We extend the notation for transforms and poses to include multiple rigid body motions observed from different coordinate frames.

Let A_t denote the coordinate frame A at time t. For any given t, the value of the transform $\mathbf{T}_{A_tB_t}$ does not depend on any external coordinate frame, as illustrated in Figure 3.2.

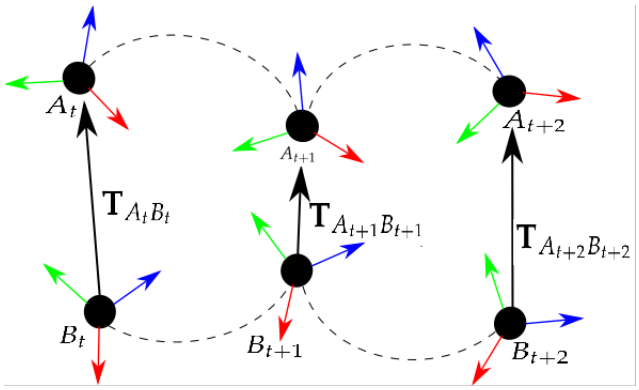


Figure 3.2: The transforms between moving reference frames A_t and B_t do not depend on any external or fixed coordinate frame, for a given time t.

Transforms between frames observed at the same time may be inverted and concatenated as expected;

$$\mathbf{T}_{A_tB_t}^{-1} = \mathbf{T}_{B_tA_t} \tag{3.11}$$

$$\mathbf{T}_{A_tC_t} = \mathbf{T}_{A_tB_t}\mathbf{T}_{B_tC_t} \tag{3.12}$$

Computing the value of the transform $\mathbf{T}_{A_iB_j}$ between two coordinate frames A, B observed at different times i, j requires the choice of an observing or *fixed* frame. We denote a transform from B_j to A_i as observed from the fixed frame C as $\mathbf{T}_{A_iB_j}^C$. This situation is illustrated in Figure 3.3

A *fixed* frame may be attached to the 'stationary' part of the scene, i.e. the world or background, or it may be attached to a moving object; here fixed refers to the fact that the observing frame does not change over time, i.e;

$$\mathbf{T}_{C_iC_j}^C = \mathbf{I}, \quad \forall i, j \tag{3.13}$$

Where $\mathbf{I}$ is the identity transform. We subsequently use the following shorthand for the case that one of the involved frames is the fixed frame;

$$\mathbf{T}_{CA_i} := \mathbf{T}_{C_iA_i}^C = \mathbf{T}_{C_jA_i}^C, \quad \forall i, j \tag{3.14}$$

Through concatenation we may express a fixed-frame dependent transformation in terms of independent transforms as follows;

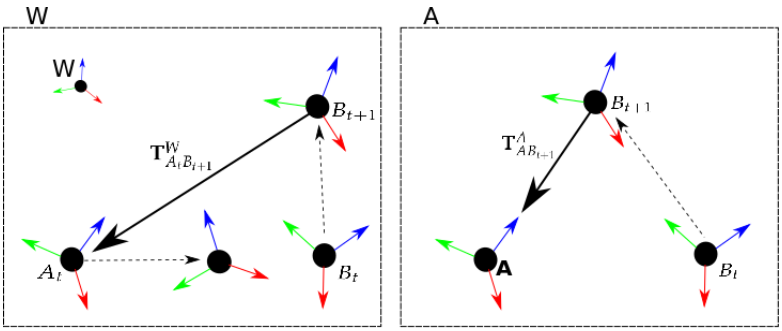


Figure 3.3: The transform between reference frames A_t and B_{t+1}, depends on the choice of observing or fixed coordinate frame. The left figure illustrates the transform as seen from an external 'world' frame W, while the right shows the observed transform from a references frame fixed to A.

$$\mathbf{T}^C_{A_i B_j} = \mathbf{T}^C_{A_i C}\mathbf{T}^C_{C B_j} = \mathbf{T}_{A_i C_i}\mathbf{T}_{C_j B_j} \tag{3.15}$$

This is particularly useful as when C refers to the fixed world frame, it allows us to compute the relative transforms of rigid body frames attached to moving objects between times i, j by establishing the transforms between those objects and the world frame at the corresponding times.

We may also thus establish inverse transforms at different times;

$$\mathbf{T}^C_{A_i B_j}{}^{-1} = \left(\mathbf{T}_{A_i C_i}\mathbf{T}_{C_j B_j}\right)^{-1} = \mathbf{T}_{B_j C_j}\mathbf{T}_{C_i A_i} = \mathbf{T}^C_{B_j A_i} \tag{3.16}$$

As well as concatenation;

$$\mathbf{T}^X_{A_i B_k}\mathbf{T}^X_{B_k C_j} = \mathbf{T}_{X_k B_k}\mathbf{T}_{B_k X_k}\mathbf{T}_{X_j C_j} = \mathbf{T}_{A_i X_i}\mathbf{T}_{X_j C_j} = \mathbf{T}^X_{A_i C_j} \tag{3.17}$$

With these operations we may now proceed to the concrete example of a moving camera observing moving objects, relative to a fixed world frame.

3.1.3 Observing rigid body motion from a moving camera

Consider C the coordinate frame attached to a camera moving relative to the static world frame W. Additionally, consider the frame O attached to an object moving both

relative to the camera frame and the world frame. We wish to compute the motion of the object in the world frame between times i and $j > i$, given the initial relative pose of the camera in the object frame $\mathbf{T}_{C_iO}$, the transform corresponding to the motion of the camera relative to the world frame $\mathbf{T}^W_{C_iC_j}$, and the motion of the camera relative to the object.

First we wish to express the motion of the camera pose in the world frame. At time i, the pose of the camera in the world frame is given as $\mathbf{T}_{WC_i}$. The transform between the camera pose from time $j > i$ to time i as seen in the world frame is then $\mathbf{T}^W_{C_iC_j}$.

We now consider the motion of the camera relative to the object. The observed relative motion is due to a combination of the motion of the camera relative to the world, as well as the motion object relative to the world. When observed in a fixed coordinate frame attached to the object, the transform of the camera from time $j > i$ to time i can be expressed as $\mathbf{T}^O_{C_iC_j}$. The overall situation is shown in Figure 3.4

We wish to compute $\mathbf{T}_{WO_j}$. Using the previously established operations, we may compute this as;

$$\begin{aligned}\mathbf{T}_{WO_j} &= \mathbf{T}_{W_jO_j} \\ &= \mathbf{T}_{W_jC_j}\mathbf{T}_{C_jO_j} \\ &= \mathbf{T}_{W_jC_j}\mathbf{T}^O_{C_jO_j} \\ &= \mathbf{T}_{WC_i}\mathbf{T}^W_{C_iC_j}\mathbf{T}^O_{C_jC_i}\mathbf{T}^O_{C_iO_j}\end{aligned} \tag{3.18}$$

We note that, because O is fixed to the object, $\mathbf{T}^O_{C_iO_j} = \mathbf{T}^O_{C_iO_i} = \mathbf{T}_{C_iO}$ which is given. The relative pose of the camera corresponding to the motion of the camera observed in the object frame, $\mathbf{T}^O_{C_iC_j}$, is also given. Thus;

$$\mathbf{T}_{WO_j} = \mathbf{T}_{W_jC_j}\mathbf{T}^O_{C_jC_i}\mathbf{T}_{C_iO} \tag{3.19}$$

Thus given estimates of the relative motion of the camera in the world frame, and the camera in the object frame, we may compute the pose of the object in the world frame at time j. The transform due to the motion of the object in the world frame is then given as;

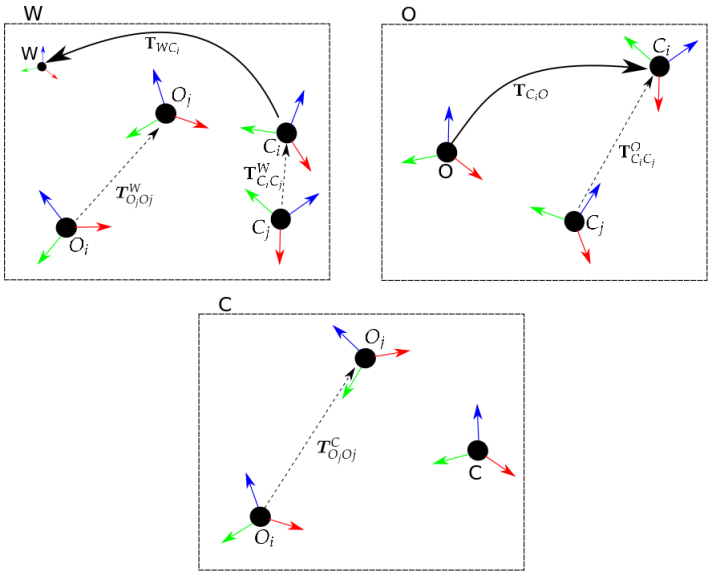


Figure 3.4: A moving object observed from a moving camera in three coordinate frames. Top left: The fixed world frame W. Top right: The frame fixed to the object, O. Bottom: The frame fixed to the camera, C. The given transforms are shown.

$$\begin{aligned}\mathbf{T}^W_{O_iO_j} &= \mathbf{T}_{O_iW}\mathbf{T}_{WO_j} \\ &= \mathbf{T}_{WO_i}{}^{-1}\mathbf{T}_{WO_j}\end{aligned} \tag{3.20}$$

By computing the sequence of transforms $\mathbf{T}_{WO_t}, t \in \{0, 1, \ldots, T\}$, we may compute the full trajectory of the object so long as it is visible in the camera. In order to do so, we apply the direct photometric alignment algorithm to estimate camera motion relative to the object, and relative to the world frame.

3.2 Direct Photometric Alignment

We now turn to the discussion of the direct photometric alignment algorithm used to estimate motion transforms. We first discuss the so-called warping function, a fundamental component of this algorithm. Subsequently we derive the direct alignment algorithm itself.

3.2.1 Warping Function

The warping function, sometimes also referred to as the reprojection function, allows the computation of the corresponding location of a point in one image, to the location of the same point in another image, given a rigid body transform.

The warping function $\tau : \mathbb{R}^2 \times \mathbb{R} \times SE(3) \to \mathbb{R}^2$ takes as arguments a point $\boldsymbol{x}_1 \in \mathbb{R}^2$ in an image $\boldsymbol{I}_1$, its corresponding depth $d \in \mathbb{R}^+$ with respect to the camera coordinates of the image, and a rigid body transform $\boldsymbol{T} \in SE(3)$.

The transform $\boldsymbol{T}$ corresponds to the transform from camera frame C_1 to camera frame C_2, relative to some fixed reference frame. This may be a static world frame, or a frame attached to a moving object, as discussed in 3.1.2. Thus $\boldsymbol{T}$ may correspond to either the motion relative to the world, or the relative motion of the camera to a moving object.

The output of the warping function is a point $\boldsymbol{x}_2$ in the coordinates of a new image $\boldsymbol{I}_2$, corresponding to the camera coordinate frame C_2. The warping function is computed as;

$$\boldsymbol{x}_2 = \tau(\boldsymbol{x}_1, d, \boldsymbol{T}) = \pi\left(\boldsymbol{T} \cdot \pi^{-1}(\boldsymbol{x}_1, d)\right) \tag{3.21}$$

Where π is the familiar perspective projection, and π^{-1} is the so-called unprojection function, which transforms a point from image coordinates to camera coordinates, given depth.

It is important to note, that in this formulation of the warping function, there is no restriction that points in $\boldsymbol{I}_1$ are uniquely mapped to points $\boldsymbol{I}_2$ as the warp is computed independently for each point. Though this is not a problem in the case that the coordinates of points in an image may take continuous values, in practice images from real cameras are made up of a relatively small number of discrete pixels, and correspondingly relatively few image points with discrete coordinates.

The warping of a set of points in $\boldsymbol{I}_1$ to the same point in $\boldsymbol{I}_2$ represents the case where an occlusion has occurred between two images; the change of viewing angle has caused elements of the scene to obscure each other, or else object motion has caused objects to move in front of one another from the camera's point of view.

It is possible to resolve these occlusions explicitly through computing which point in $\boldsymbol{I}_1$ is warped to the point with the smallest depth in $\boldsymbol{I}_2$, thus establishing which point is in 'front'. In this thesis we do not consider such occlusions, under the assumption that the relative motions between frames are sufficiently small that only a small number of points may be occluded.

Having established a function for transforming image points between images for a given relative transform, we proceed to give details on estimating a rigid body transform from an image sequence via direct alignment.

3.2.2 Photometric Consistency

The foundation of the direct alignment algorithm is the photometric consistency assumption. Photometric consistency states that, for a given point in the world viewed in two different images, we should expect the corresponding points in each image to have the same intensity. Stated more formally;

$$I_1(\boldsymbol{x}_1) = I_2(\tau(\boldsymbol{x}_1, d_1, \boldsymbol{T})) = I_2(\boldsymbol{x}_2), \forall \boldsymbol{x}_1 \in \boldsymbol{I}_1 \tag{3.22}$$

Where $\boldsymbol{T} \in SE(3)$ is the rigid body transform from the coordinate frame of the camera corresponding to the image $\boldsymbol{I}_1$, to the coordinate frame corresponding to the image $\boldsymbol{I}_2$, and τ is the so-called 'warping' or reprojection function discussed in Section 3.2.1.

Note that this formulation of photometric consistency requires that the depth d_1 corresponding to each $\boldsymbol{x}_1 \in \boldsymbol{I}_1$ is either estimated, or otherwise known in advance. We subsequently refer to an image frame that has corresponding depth information as a *keyframe*, and a frame that contains only intensity information as an *image frame*, throughout this thesis.

In later sections, we will also assign segmentation information to the keyframes.

We note is that because real image frames are of finite size, it is possible for pixels in $\boldsymbol{I}_1$ to be warped to positions outside the bounds of $\boldsymbol{I}_2$, and thus no determination of the photometric consistency can be made in such cases. Various approaches have been proposed to deal with this problem, however in this thesis we weight the contribution of pixels which at any time are warped out of bounds (including during the optimization

procedure formulated in later sections) to 0, by setting the corresponding photometric residual to 0.

This has the undesirable property that the photometric residual can be globally minimized by warping all pixels out of bounds. However, we did not observe that optimization converged to this global minimum in any of our experiments.

3.2.3 Probabilistic formulation

In reality, photometric consistency does not always hold, due to a variety of factors. In particular, noise at the image sensor may result in a photometric residual, expressed as the difference in intensities for the same point between the two images;

$$r\left(\boldsymbol{x_1}, d_1, \boldsymbol{T}\right) \coloneqq \boldsymbol{I}_2\left(\tau\left(\boldsymbol{x}_1, d_1, \boldsymbol{T}\right)\right) - \boldsymbol{I}_1\left(\boldsymbol{x}_1\right) \tag{3.23}$$

Let r_i be the random valued residual corresponding to a point $\boldsymbol{x}_i \in \boldsymbol{I}_1$, and $\boldsymbol{r}$ the random-valued vector with elements corresponding to the residuals of each point in $\boldsymbol{I}_1$, $\boldsymbol{r} = \left(r_1, \ldots, r_n\right)^\top$.

Assuming that the elements of $\boldsymbol{r}$ are independent and identically distributed, and taking d as given;

$$p\left(\boldsymbol{r}|\boldsymbol{T}\right) = \prod_i p\left(r_i|\boldsymbol{T}\right) \tag{3.24}$$

The aim of the direct alignment algorithm is to find a transform such that the posterior probability $p\left(\boldsymbol{T}|\boldsymbol{r}\right)$ is maximal;

$$\boldsymbol{T}_{MAP} = \arg\max_{\boldsymbol{T}} p\left(\boldsymbol{T}|\boldsymbol{r}\right) \tag{3.25}$$

By Bayes' rule;

$$p\left(\boldsymbol{T}|\boldsymbol{r}\right) = \frac{p\left(\boldsymbol{r}|\boldsymbol{T}\right) p\left(\boldsymbol{T}\right)}{p\left(\boldsymbol{r}\right)} \tag{3.26}$$

By (3.24) and (3.26), and noting that $p\left(\boldsymbol{r}\right)$ is constant in $\boldsymbol{T}$, the maximum posterior probability becomes;

$$\boldsymbol{T}_{MAP} = \arg\max_{\boldsymbol{T}} \prod_i p\left(r_i|\boldsymbol{T}\right) p\left(\boldsymbol{T}\right) \tag{3.27}$$

Taking the negative log-likelihood and minimizing gives;

$$\boldsymbol{T}_{MAP} = \arg\min_{\boldsymbol{T}} -\sum_i \log p\left(r_i|\boldsymbol{T}\right) - \log p\left(\boldsymbol{T}\right) \tag{3.28}$$

The term $\log p\left(\boldsymbol{T}\right)$ may be regarded as a prior on the transform. The prior might be set from a motion model, or similar additional information about the typical characteristics of the observed motions that may give rise to $\boldsymbol{T}$. If we assume all transforms are equally likely, then this term becomes a constant and does not contribute to the minimizer. Hence, it can be discarded.

Taking derivatives with respect to $\boldsymbol{T}$ and setting to zero gives a minimum at;

$$\sum_i \frac{\partial \log p\left(r_i|\boldsymbol{T}\right)}{\partial \boldsymbol{T}} = \sum_i \frac{\partial \log p\left(r_i|\boldsymbol{T}\right)}{\partial r_i}\frac{\partial r_i}{\partial \boldsymbol{T}} = 0 \tag{3.29}$$

We define $w(r_i) = \partial \log p\left(r_i|\boldsymbol{T}\right)/\partial r_i \cdot 1/r_i$, giving;

$$\sum_i \frac{\partial r_i}{\partial \boldsymbol{T}} w(r_i) r_i = 0 \tag{3.30}$$

Equation 3.30 is a statement of a first order stationary point, and hence the minima of a quadratic system. Hence, the direct alignment algorithm reduces to minimizing a weighted least-squares system;

$$\boldsymbol{T}_{MAP} = \arg\min_{\boldsymbol{T}} \sum_i w\left(r_i\right)\left(r_i\left(\boldsymbol{T}\right)\right)^2 \tag{3.31}$$

The choice of distribution for $\boldsymbol{r}$ determines the so-called weighting function $w\left(r_i\right)$. For example, a Gaussian distribution results in a constant weighting. We examine this aspect of the algorithm in more depth in subsequent sections.

Next we discuss the approach to minimizing the least-squares system.

3.2.4 Non-Linear Weighted Least Squares Energy Minimization

The least squares system in equation 3.31 is non-linear, as the warping function τ is non-linear and therefore does not admit a closed-form solution as in linear least-squares systems. However, a solution may be obtained iteratively through application of the

Gauss-Newton method.

The Gauss-Newton method finds a solution to the non-linear least-squares problem by iteratively computing the solution of linear least-squares problems through linearizion of the non-linear system. The solution of each linear least-squares sub-problem is then applied as an increment to the estimate of $\boldsymbol{T}$, until convergence. Since the value of the residual r_i is recomputed at each iteration, the weight $w(r_i)$ must also be recomputed at each iteration. Thus we consider the Gauss-Newton algorithm for iteratively re-weighted least squares.

Gauss-Newton Algorithm for Iteratively re-weighted least squares.

As discussed in Section 3.1.1, performing optimization directly on $\boldsymbol{T}$ is problematic, and we are better served by using the corresponding Lie algebra, represented as a vector $\hat{\boldsymbol{\xi}} \in \mathbb{R}^6 = \log(\boldsymbol{T})$, as a model parametrization. Notably, the use of a Lie algebra for performing iterative optimization is justified since we perform a local linearization about the parameters at the current iteration.

To make this parametrization explicit, we restate equation 3.31 as;

$$\hat{\boldsymbol{\xi}}_{MAP} = \arg\min_{\hat{\boldsymbol{\xi}}} \sum_i w(r_i) \left(r_i\left(\hat{\boldsymbol{\xi}}\right)\right)^2 \tag{3.32}$$

Let $\hat{\boldsymbol{\xi}}_k$ be the estimate of $\hat{\boldsymbol{\xi}}$ at the current iteration, and $\hat{\boldsymbol{\xi}}_{k+1}$ be the estimate at the next iteration. Let $\hat{\boldsymbol{\xi}}_{k+1} = \Delta\boldsymbol{\xi}_k \boxplus \hat{\boldsymbol{\xi}}_k$, where $\Delta\boldsymbol{\xi}_k$ is the computed increment to the estimate at the k^{th} iteration. Here the operator $\boxplus : \mathfrak{se}(3) \times \mathfrak{se}(3) \mapsto \mathfrak{se}(3)$ is an extension of the composition of rigid body transforms to their lie algebra representations, and is computed as;

$$\Delta\boldsymbol{\xi}_k \boxplus \hat{\boldsymbol{\xi}}_k = \log\left(\exp\left(\Delta\boldsymbol{\xi}_k\right) \cdot \exp\left(\hat{\boldsymbol{\xi}}_k\right)\right) \tag{3.33}$$

This is referred to as the left-compositional formulation [ESC14][KSC13].

At each iteration of the Gauss-Newton algorithm, the residual at each pixel becomes;

$$r_i\left(\Delta\boldsymbol{\xi}_k \boxplus \hat{\boldsymbol{\xi}}_k\right) = I_2\left(\tau\left(\Delta\boldsymbol{\xi}_k \boxplus \hat{\boldsymbol{\xi}}_k\right)\right) - I_1\left(\boldsymbol{x}_i\right) \tag{3.34}$$

Linearization of the residual about $\Delta\boldsymbol{\xi}_k = \mathbf{0}$ via first-order Taylor expansion gives;

$$r_{i,\text{lin}}\left(\Delta\boldsymbol{\xi}_k \boxplus \hat{\boldsymbol{\xi}}_k\right)\big|_{\Delta\boldsymbol{\xi}_k=\mathbf{0}} = r\left(\hat{\boldsymbol{\xi}}_k, \boldsymbol{x}_i\right) + \left.\frac{\partial r\left(\tau\left(\Delta\boldsymbol{\xi}_k \boxplus \hat{\boldsymbol{\xi}}_k\right)\right)}{\partial \Delta\boldsymbol{\xi}}\right|_{\Delta\boldsymbol{\xi}_k=\mathbf{0}} \Delta\boldsymbol{\xi}_k \tag{3.35}$$

$$= r\left(\hat{\boldsymbol{\xi}}_k, \boldsymbol{x}_i\right) + \boldsymbol{J}_i \Delta\boldsymbol{\xi}_k \tag{3.36}$$

Where $\boldsymbol{J}_i \in \mathbb{R}^6$ is the row of the Jacobian matrix $\boldsymbol{J} \in \mathbb{R}^{n\times 6}$ corresponding to the i^{th} point.

Inserting into the optimality condition given by 3.30 gives;

$$\boldsymbol{J}^\top \boldsymbol{W} \boldsymbol{J} \Delta\boldsymbol{\xi}_k = -\boldsymbol{J}^\top \boldsymbol{W} \boldsymbol{r}\left(\hat{\boldsymbol{\xi}}_k\right) \tag{3.37}$$

We note that the term $\boldsymbol{J}^\top \boldsymbol{W} \boldsymbol{J}$ approximates the matrix of second derivatives (the Hessian) of the nonlinear least-squares system.

Here $\boldsymbol{W}$ is the diagonal weight matrix, where each diagonal entry corresponds to $w(r_i)$, the weight at the corresponding residual given by the weight function. The choice of weight function is important for the performance of the algorithm, as it determines the influence of outliers on the found minimum, and consequently, the residual value at the outliers which is important for our approach to moving object detection. We discuss the choice of weight function and other robustness considerations in Section 3.2.4.

We obtain an increment $\Delta\boldsymbol{\xi}_k$ by;

$$\Delta\boldsymbol{\xi}_k = \left(\boldsymbol{J}^\top \boldsymbol{W} \boldsymbol{J}\right)^{-1} \left(-\boldsymbol{J}^\top \boldsymbol{W} \boldsymbol{r}\left(\hat{\boldsymbol{\xi}}_k\right)\right) \tag{3.38}$$

Which is the solution of a linear system of equations. We note that though finding $\Delta\boldsymbol{\xi}$ appears to require a computationally expensive matrix inversion and several large matrix multiplications, in practice the linear system has only dimension 6, and can therefore be solved both quickly and accurately using modern linear algebra solvers. In addition, we can exploit the fact that we can compute each element of $\left(\boldsymbol{J}^\top \boldsymbol{W} \boldsymbol{J}\right)$ and $-\boldsymbol{J}^\top \boldsymbol{W} \boldsymbol{r}\left(\hat{\boldsymbol{\xi}}_k\right)$ in parallel using an accumulator approach.

The Gauss-Newton algorithm is iterated until convergence. Convergence is typically determined as either changes in the magnitude of the residual vector $\boldsymbol{r}$, or the magnitude of the vector representation of the computed increment $\Delta\boldsymbol{\xi}$ becoming sufficiently

small. The Gauss-Newton method converges only to the closest local minimum, and may diverge. Therefore it is important that the initial estimate for the transform, $\hat{\xi}_I$, is close to the true global minimum.

In this thesis, we initialize with the most recently computed estimate for the transform, under the assumption that all relative motions are sufficiently small that this represents a good initialization.

The direct photometric alignment algorithm is illustrated in figure 3.5.

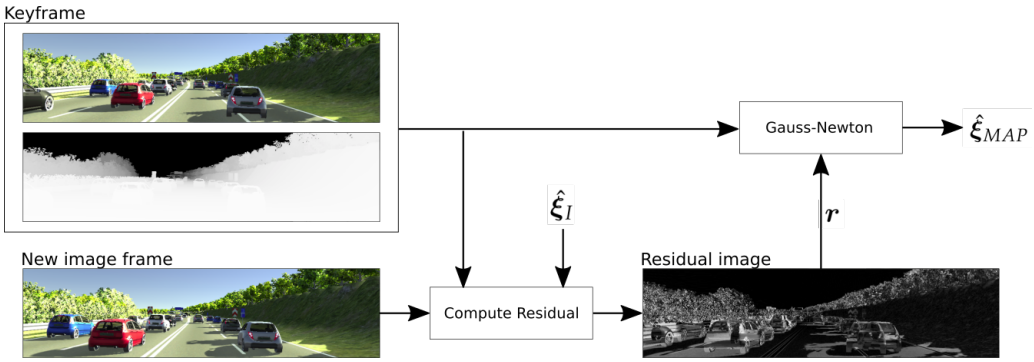


Figure 3.5: Overview of the direct photometric alignment algorithm. The residual is recomputed in each iteration.

Several extensions to the basic Gauss-Newton method have been developed, including the Levenberg-Marquardt algorithm which produces a more robust optimization at the expense of speed of convergence. We use only the presented form of the Gauss-Newton algorithm throughout this thesis, in order to highlight the results and effectiveness of the base form of the presented algorithm, rather than the influence of any particular mathematical approach.

Robustness Considerations

Several authors [KSC13][ESC14][EKC18] have shown that significant improvements to the accuracy of the transform estimates from Gauss-Newton iteratively re-weighted least squares can be achieved through relatively simple extensions. These include improving robustness to outlying residuals through the use of an appropriate weight function, and improved initialization through the use of a coarse-to-fine scheme.

Robust Weights

The choice of weight $w(r_i)$ function in equation 3.31 reflects the distribution of the photometric residuals. Assuming Normally distributed photometric residuals results in a constant weight function, i.e. $w(r_i) = k$. However, it has been shown empirically that photometric residuals for most sequences have a large proportion of outliers [KSC13].

To mitigate the influence of these outlying residuals, we employ the so-called Huber weight function [Hub11][NLD11]. It is defined as;

$$w_{\text{huber}}(r_i) = \begin{cases} 1 & |r_i| \leq k \\ \frac{k}{|r_i|} & else \end{cases} \tag{3.39}$$

Applying the Huber weight function results in quadratic contributions from small residuals, and only linear contributions from outliers. Assuming normally distributed photometric residuals with outliers, the parameter k is set to 1.345 [Wer+09]. We use the Huber weight function for performing direct photometric alignment throughout this thesis.

Coarse to fine optimization

Coarse to fine optimization refers to performing direct photometric alignment on a downsampled, lower resolution image, in order to initialize the transform estimate for the full-resolution image. This can be accomplished recursively, in a so-called pyramid of downsampled images. Such coarse to fine optimization has been shown to be effective in increasing the convergence radius of the Gauss-Newton optimization [ESC14].

Each level of the pyramid is down-sampled by halving the resolution in each dimension for each level. For example, an image with resoltuion 640x480 would be down-sampled to 320x240 in the first level, 160x120 in the second, and so on. Four levels have been shown to be effective for 640x480 images. Downsampling is accomplished through bilinear interpolation in both image and intensity images, for both the keyframe and the target image frame.

The use of such a coarse-to-fine approach introduces a tradeoff when tracking dynamic objects. Any given object represents only a fraction of the pixels in a given

image of the scene as a whole. It may therefore not be visible at all at higher pyramid levels, and hence cannot be tracked there. We therefore use a number of pyramid levels that reflect the fraction of pixels corresponding to the object of interest.

Let n be the number of pyramid levels used for the full scene of P pixels. Then if p_o pixels in the keyframe image correspond to object o, we use $n_o = \lceil n\frac{p}{P} \rceil$ levels when performing direct alignment to estimate the transform corresponding to the motion of object o.

Having defined the direct photometric alignment approach to estimating rigid body transforms between two images, we next turn to finding points belonging to moving objects through segmenting the keyframe to inlier and outlier points.

3.3 Binary Image Segmentation with Conditional Random Fields

Key to the function of the algorithm is the determination of which subsets of points in the keyframe belong to independently moving objects. We approach finding these subsets through computing a segmentation of the discrete pixels in the keyframe using photometric outliers.

3.3.1 Motion and photometric residuals

Consider independently moving objects O_1, O_2 with attached coordinate frames, observed by a camera C. Let $\boldsymbol{T}_C^1$ correspond to the relative motion of the camera in the frame of O_1, and $\boldsymbol{T}_C^2$ correspond to the relative motion of the camera in the frame of O_2, between two time steps t_1, t_2.

Let the images observed by the camera at t_1 and t_2 be the intensity image component of the keyframe $\boldsymbol{K}$ and the new image $\boldsymbol{I}$ respectively. We We assume that, given sufficiently textured objects and a sufficiently textured background, photometric consistency as stated in 3.22 will be violated if the transform corresponding to the motion of O_2 is applied to observations of points on O_1. That is for some points $\boldsymbol{x}_1$ in image $\boldsymbol{I}_1$ corresponding to points on O_1;

$$\exists \boldsymbol{x}_1 \mid \boldsymbol{K}\left(\boldsymbol{x}_1\right) \neq \boldsymbol{I}\left(\tau\left(\boldsymbol{x}_1, d_1, \boldsymbol{T}_C^{\boldsymbol{2}}\right)\right) \tag{3.40}$$

This situation is illustrated in figure 3.6. A similar argument holds for points on O_2.

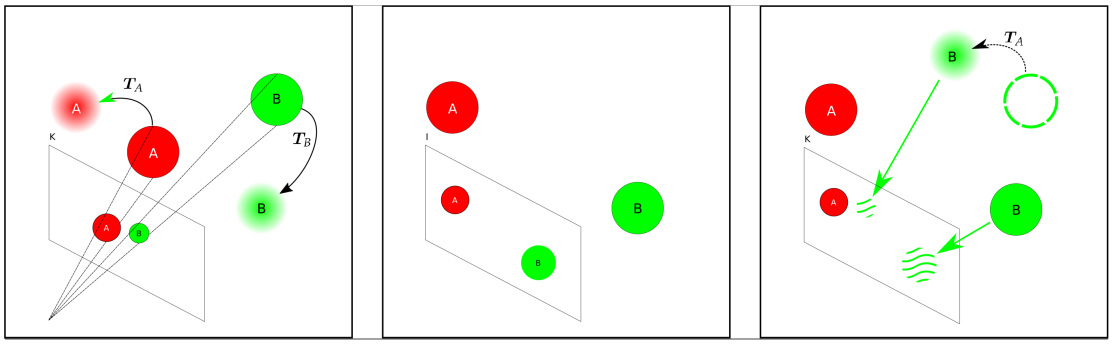


Figure 3.6: Independently moving objects A and B are observed in the image space of the keyframe K. **Left:** The initial positions of A and B relative to the camera, and their subsequent motions. **Middle:** A new image frame I reflecting the true positions of A and B after undergoing independent motion. **Right:** A warping function is applied to points on B, taking the transform corresponding to the motion of A as an argument. Points belonging to B are warped such that the relative position of B to A remains the same. This gives rise to a high photometric residual (visualized by wavy lines), both at the now erroneous projection of the points belonging to B, as well as at the projection of the true position of B.

It follows that the photometric residual as stated in 3.23 will be large for pixels in the keyframe where the supplied transform does not match the motion of points lying on an independently moving object. We subsequently refer to rigid body transforms that give rise to large photometric residuals as *inconsistent* transforms, and their motions as inconsistent motions.

We may therefore use the photometric residual as a cue that an estimated transform does not match with the relative motion of some part of the scene, and hence that outlying photometric residuals indicate an inconsistently moving object.

To do so we construct the *residual image* $\boldsymbol{R} \in \mathbb{R}^{n \times m}$. The value of each element $r_i \in \boldsymbol{R}$ for a given transform $\boldsymbol{T}$ and new intensity image $\boldsymbol{I}$ is computed as;

$$r_i \coloneqq \boldsymbol{I}\left(\tau\left(\boldsymbol{x}_i, d, \boldsymbol{T}\right)\right) - \boldsymbol{K}\left(\boldsymbol{x}_i\right) \tag{3.41}$$

Using the photometric residual to detect inconsistently moving objects for a given transform and new image amounts to segmenting the resulting residual image into pixels with inlying (likely) residuals, or outlying (unlikely) residuals.

A simple approach to segmenting outliers is to choose a threshold, above which the residual is considered too high to be due to random noise at the sensor, and below which the estimated transform explains the observation sufficiently well.

However, this simple formulation does not capture dependencies between neighboring pixels, which are likely to belong to the same object, and does not regularize for outliers that are caused by random noise.

Instead, we formulate the outlier segmentation problem in terms of a binary labeling problem on a conditional random field, which models the probability of each pixel in the image being an outlier as depending on the observed residual at that pixel, as well as the labels of neighboring pixels. Subsequently, the segmentation is performed via graph cuts.

3.3.2 Conditional Random Fields for Segmentation

Let $\boldsymbol{Y} \in \{0,1\}^{n \times m} \in \mathcal{Y}$ be a binary labeling of each of the $n \times m$ pixels in the residual image, where a label y_i for pixel $\boldsymbol{x}_i$ of 0 corresponds to an inlier, and a label of 1 corresponds to an outlier.

We wish to infer a labeling, given observations on the residual image and encoding pair-wise dependencies on neighboring pixels. This requires that we are able to express the conditional distribution $p(\boldsymbol{Y}|\boldsymbol{R})$. A natural representation of such a distribution is a Conditional Random Field [KFB09].

A conditional random field is an undirected graph encoding conditional dependencies. It is commonly used in computer vision as a useful representation for probabilistic inference on images, since it naturally encodes both dependencies on observations as well as on neighbors.

The conditional random field consists of a set of edges and nodes. Each node corresponds to either a target variable, in our case the binary inlier/outlier label y_i, or an observed variable, in our case the observed residual r_i. Each edge represents a conditional dependency between variables. A simple example is illustrated in figure 3.7.

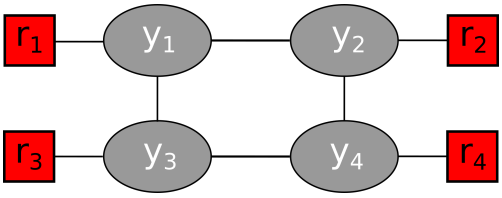


Figure 3.7: A simple conditional random field represented as a graph. Each node y_i has an edge representing the conditional dependence on the residual r_i, as well as neighboring pixels.

The edges define a set of factors with associated *potentials* $\phi(y_i, r_i)$ over labels and observations, and $\phi(y_i, y_j)$ over pairs of labels. The distribution encoded by the graph is then;

$$\begin{aligned} p(\boldsymbol{Y}|\boldsymbol{R}) &= \frac{1}{Z(\boldsymbol{R})}\tilde{p}(\boldsymbol{Y}|\boldsymbol{R}) \\ \tilde{p}(\boldsymbol{Y}|\boldsymbol{R}) &= \prod_{i}^{n\times m}\phi(y_i, r_i)\prod_{j\in \mathrm{N}(\boldsymbol{x}_i)}\phi(y_i, y_j) \\ Z(\boldsymbol{R}) &= \sum_{\boldsymbol{Y}\in\mathcal{Y}}\tilde{p}(\boldsymbol{Y}|\boldsymbol{R}) \end{aligned} \tag{3.42}$$

Where $\mathrm{N}(\boldsymbol{x}_i)$ refers to the index of neighbors of pixel $\boldsymbol{x}_i$. In our model, these are the south and east neighbors, resulting in factors for each pixel in the 4-neighborhood of any given pixel. Other neighborhoods, for example the 8-neighborhood which also includes a pairwise dependence on the diagonally neighboring pixels, are also possible.

A key feature of the conditional random field representation is that it allows us to avoid encoding a joint distribution $P(\boldsymbol{Y}, \boldsymbol{R})$ which may be unknown or which may not have a simple parametric form.

This is advantageous in the case of the segmentation problem since it is difficult to precisely express the joint distribution between segmentation and residual image. Thus

we may construct the potentials according to domain knowledge and the particular features of the specific problem.

We wish to find the most likely labeling, given the observed residual image; i.e. we wish to find the labeling which gives the maximum posterior likelihood over all labellings;

$$\boldsymbol{Y}_{MAP} = \arg\max_{\boldsymbol{Y}} p(\boldsymbol{Y}|\boldsymbol{R}) = \arg\max_{\boldsymbol{Y}} \frac{1}{Z(\boldsymbol{R})}\tilde{p}(\boldsymbol{Y}|\boldsymbol{R}) = \arg\max_{\boldsymbol{Y}} \tilde{p}(\boldsymbol{Y}|\boldsymbol{R}) \tag{3.43}$$

In general evaluating each possible labeling to find the maximum is not computationally tractable for even small residual images, since for an image of $n \times m$ pixels there are $2^{(n\times m)}$ possible labellings. By taking the negative log, we may re-cast the problem in terms of an energy minimization over a sum;

$$\arg\max_{\boldsymbol{Y}} \tilde{p}(\boldsymbol{Y}|\boldsymbol{R}) = \arg\min_{\boldsymbol{Y}} -\log\left(\tilde{p}(\boldsymbol{Y}|\boldsymbol{R})\right) \tag{3.44}$$

$$\arg\min_{\boldsymbol{Y}} -\log\left(\tilde{p}(\boldsymbol{Y}|\boldsymbol{R})\right) = \arg\min_{\boldsymbol{Y}} \sum_{i}^{n\times m} -\log\left(\phi(y_i, r_i)\right) + \sum_{j\in \mathrm{N}(\boldsymbol{x}_i)} -\log\left(\phi(y_i, y_j)\right) \tag{3.45}$$

Let $E_i(y_i|r_i) = -\log\left(\phi(y_i, r_i)\right)$, referred to as the unary energy terms, and let $E_{i,j}(y_i, y_j) = -\log\left(\phi(y_i, y_j)\right)$, referred to as the pairwise terms. We then have;

$$\boldsymbol{Y}_{MAP} = \arg\max_{\boldsymbol{Y}} p(\boldsymbol{Y}|\boldsymbol{R}) = \arg\min_{\boldsymbol{Y}} \sum_{i}^{n\times m} E_i(y_i|r_i) + \sum_{j\in \mathrm{N}(\boldsymbol{x}_i)} E_{i,j}(y_i, y_j) \tag{3.46}$$

It has been shown that so long as each pairwise term is *regular*, then it is possible to find the global minimum of this energy in polynomial time for binary variables y_i, by solving the min-cut max-flow problem for an appropriately constructed graph [GPS89] [BK04].

3.3.3 Segmentation Energies

Let $\mathbb{B}^2$ denote the set of 2-dimensional binary vectors. The terms $E_{i,j}\colon \mathbb{B}^2 \mapsto \mathbb{R}$ are regular iff;

$$E_{i,j}(0,0) + E_{i,j}(1,1) \leq E_{i,j}(1,0) + E_{i,j}(0,1) \tag{3.47}$$

It remains to choose potential functions, and hence energies, such that the condition is met, while modeling the outlier likelihood and pairwise conditional dependence.

The unary terms of the energy correspond to the influence of observations at each pixel. As per Section 3.3.1, we wish to use the photometric residual as a cue to determine the inlying and outlying pixels. Hence, we wish to choose potential functions and energies such that they are dependent on the pixel-wise photometric residuals r_i.

Assuming that the photometric residuals at inlying pixels are approximately Gaussian distributed with mean 0; $r_i \sim \mathcal{N}(0, \sigma)$, a natural choice of unary potential function for inliners, $\phi_i(0, r_i)$, is the Gaussian probability mass function, normalized such that it assumes values in $(0, 1]$;

$$\phi_i(0, r_i) = \sigma\sqrt{2\pi} \times p(r_i) = e^{-r_i^2/2\sigma^2} \tag{3.48}$$

We choose $\phi_i(1, r_i) = 1 - \phi_i(0, r_i)$. The unary energies are then;

$$E_i(0, r_i) = -\log(\phi_i(0, r_i)) = r_i^2/2\sigma^2 \tag{3.49}$$

$$E_i(1, r_i) = -\log(1 - \phi_i(0, r_i)) \tag{3.50}$$

We note that the unary energies are also parametrized by σ, corresponding to the unknown standard deviation of the distribution of inlying photometric residuals. This value may be determined empirically, or by characterizing sensor noise in the image formation process. We investigate the influence of the choice of sigma in chapter4.

The pairwise energy corresponds to the dependence between neighboring pixel labels. We choose the pairwise energies according to the Pott's model;

$$E_{i,j}(y_i, y_j) = w[\![y_i \neq y_j]\!] \tag{3.51}$$

Where $w \geq 0$ is a freely chosen weighting factor that controls the degree of dependence of neighbors, and;

$$[\![y_i \neq y_j]\!] = \begin{cases} 0 & y_i = y_j \\ 1 & else \end{cases} \tag{3.52}$$

It is clear that the resulting energy is modular, since $E_{i,j}(0,0) = E_{i,j}(1,1) = 0 \leq E_{i,j}(1,0) = E_{i,j}(0,1) = w$.

For modular energies it is possible to find a binary segmentation corresponding to the global minimum of the energy in polynomial time, using graph cuts.

3.3.4 Graph Cuts for Segmentation

For a given graph $\mathcal{G}$ consisting of a set of nodes $\mathcal{V}$ and a set of (not necessarily directed) edges between nodes $\mathcal{E}$, a graph cut between a source node $S \in \mathcal{V}$ and a sink node $T \in \mathcal{V}$ partitions the graph into disjoint sets of edges $\mathcal{E}_1, \mathcal{E}_2$ and nodes, such that S is not reachable from T, and vise-versa. This is accomplished by "cutting" edges, that is removing them from the set $\mathcal{E}$.

Given a weighted graph such that each edge $e_i \in \mathcal{E}$ has an assigned weight $w_i \in \mathbb{R}^+$, a minimum cut is then the graph cut for which the sum of the weights of the removed edges is minimum.

Several efficient algorithms for finding the minimum cut have been discovered, particularly through the solution of the dual maximum-flow problem [GPS89]. It is possible to construct a graph such that the cost of a graph cut corresponds to the resulting energy of a given binary segmentation. Thus the minimum cut corresponds to the minimum segmentation energy .

The construction of the corresponding graph for regular energies is somewhat involved and out of scope for this thesis. Here we give a high-level outline. The interested reader may find details of the graph construction algorithm used in this thesis in pp.591 of [KFB09].

We construct the graph as follows; Instantiate a source node S corresponding to the inlier segment, and a sink node T corresponding to the outlier segment. We will compute the minimum cut between these two nodes.

For each pixel p_i we wish to label, we create a node v_i. We construct a directed edge from S to each node v_i, having weight computed from the unary energy terms taking y_i as argument. We also construct a direct edge from the node v_i to the sink T, with a

weight also computed from the unary energy terms.

To account for the pair-wise terms, we add a directed edge between v_i and all nodes corresponding to neighboring pixels $v_j \in \mathcal{N}(v_i)$, with weight computed from the unary and pairwise terms taking y_i and y_j as arguments. Note that this means that each node will have one directed edge from itself to each neighbor, and one from each neighbor to the node. An example graph is illustrated in figure 3.8

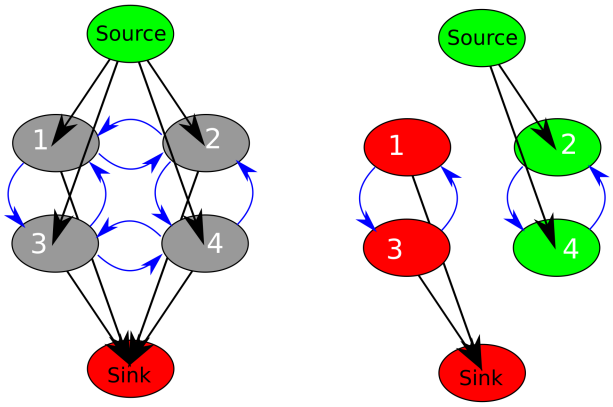


Figure 3.8: An example graph construction and resulting cut. **Left:** Directed weighted edges are constructed from the source node S to each node, and from each node to the sink node T. Edges are also constructed from each node to each of its neighbors. **Right:** The result of a graph cut. Nodes 1 and 3 are associated with the sink T, while nodes 2 and 4 are associated with S. These associations correspond to the resulting binary segmentation.

We employ the Boykov-Kolmogorov MaxFlow algorithm [BK04] to compute the minimum cut between S and T in polynomial time. After applying the computed cut, nodes reachable from S are labeled inlier and assigned to the set $\mathcal{I}$, while those from which T is reachable are labeled outlying, and assigned to the set $\mathcal{O}$.

We may now segment the residual image, and hence the keyframe, into inlying and outlying pixels, given an image and a rigid body transform, in polynomial time. The outlying pixels provide cues to regions of the keyframe for which the supplied transform may not be consistent with the true relative motion.

However, photometric outliers may be caused by effects other than relative motion, and the outlier pixels may not correspond to all points with an inconsistent relative motion. At most we may say that the observations on outlying points are not sufficiently explained by the supplied transform. This gives rise to ambiguities in the segmentation that we must resolve before we may correctly estimate the relative motion of all observed points.

In order to resolve these ambiguities, we turn to instance segmentation using a machine learning approach.

3.4 Disambiguation through Semantic Instance Segmentation

In the previous section we have established an approach for finding an outlying set of pixels in the keyframe which have photometrically inconsistent observations in a new image frame, for a given motion transform. These outliers may occur due to several phenomena, including;

- Points belonging to an object undergoing a motion is inconsistent with the supplied transform.
- Points that have become occluded in the new image.
- Points that do not conform to model assumptions such as constant brightness, due to illumination changes.

Additionally, not all points belonging to an object undergoing an inconsistent motion may be segmented as outliers. For example, a point in the interior of a low-texture region of an object may retain the same intensity between frames, despite points on the edge of object resulting in outliers. Some of these situations are illustrated in figure 3.9

Thus an ambiguity arises; though we can identify outlying pixels for which observations are definitively not explained by the supplied transform, we cannot conclusively determine the set of outliers which are due to inconsistent motions, or those outliers which are due to other phenomena such as occlusion.

This ambiguity is not generally resolvable via photometric cues alone. While it is possible to add additional information to the segmentation energy, such as geometry

Figure 3.9: Illustration of ambiguous photometric residuals. **(a)**: Image of a moving object as it appears in the keyframe. **(b)**: Images of the same object in a new image frame. It has moved right to left relative to the camera. **(c)**: The resulting residual image. Brighter pixels indicate a greater magnitude of the residual at that pixel. Pixels at the rear of the object have a high photometric residual because the motion of the object has decluded background foliage - these will be correctly segmented by the conditional random field. Pixels to the left of the object have a high photometric residual because they have become occluded, and will be incorrectly segmented. Pixels at the interior of the van do not have sufficient texture, and will not be segmented as outliers by the CRF model. **(d)**: An instance segmentation captures all pixels belonging to the object.

cues as in [SB15], or second-order intensity information such as the image gradient as suggested in [EKC18], this additional information is either not available to us for each frame, or else does not sufficiently resolve the ambiguity.

3.4.1 Machine Learning for Semantic Instance Segmentation

In recent years, machine learning approaches using convolutional neural networks, as in Mask-RCNN [He+17], have shown promising results in the so-called *dense instance segmentation* task.

In this task, given an image consisting of a set of discrete pixels $\mathcal{P}$, each pixel $p \in \mathcal{P}$ is assigned an instance label l corresponding to a single object in the scene, or else the null label (here denoted as $\emptyset$) in the case that the pixels are not associated with any particular object.

The subset of pixels which are all assigned a common label l may be referred to as an *instance* of label l.
Each instance label is also often associated with a semantic class, such as *person*, *car*, *bus*

or similar labels. Thus it is possible to associate subsets of pixels with particular objects in the scene. Where pixels are assigned $\emptyset$, it is assumed they do not correspond to any objects of a known semantic class. An example segmmentation result is shown in figure 3.10.

Figure 3.10: Segmentation example on an image from the KITTI dataset, using the network in [He+17]. Only the "car" class is shown. Colours correspond to instance labels.

We use instance segmentation to resolve the ambiguities described in the previous section, and hence to identify pixels in the keyframe corresponding to points on independently moving objects. Though there are several possible approaches which may be investigated in future work, we favor simplicity. The key assumption is that the motion of the object corresponding to each instance can be explained by a single rigid body motion.

We first segment the keyframe using a neural network approach, such that each pixel is assigned a label $l \in \{l_0, l_1, \cdots l_n, \emptyset\}$. Note that the dense instance segmentation need only be performed once per keyframe. This assigns an instance label for each pixel in the outlier set $\mathcal{O}$.

Outlier points that are assigned the null label $\emptyset$ cannot be associated with any object,

and hence are not associated with a particular motion. Often, these pixels correspond to points that have become occluded due to relative motion, or due to one object moving in front of another.

Let $|O_l|$ be the number of outlying pixels assigned the non-null label l. Let $|l|$ be the total number of pixels in the keyframe that are assigned the label l. If the fraction $|O_l|/|l|$ is greater than some chosen threshold t, i.e. if a given instance contains a sufficiently large fraction of outlying points, we take this as a cue that an independently moving object is present, for which we will need to estimate a separate rigid body transform. Labels for which the threshold is exceeded make up the *active label set* $\mathcal{L}$. The overall procedure is illustrated in figure 3.11.

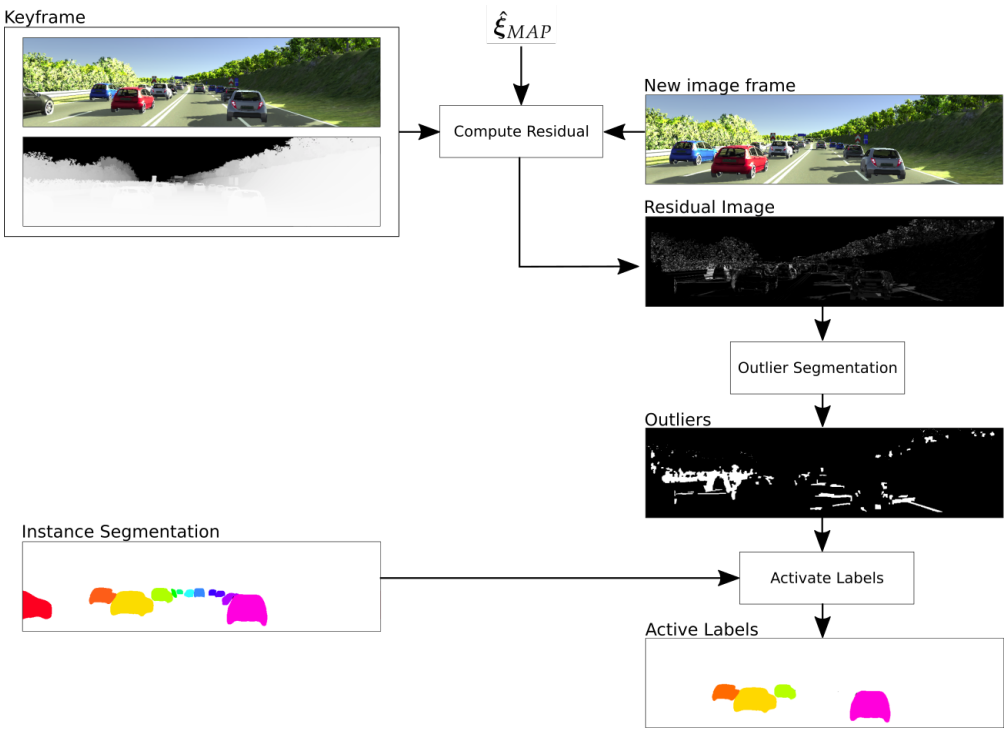


Figure 3.11: Illustration of the algorithm for activating instance segments. We perform outlier segmentation over the entire keyframe, and subsequently activate instance segments based on the number of outlying pixels.

We now have a method for identifying independently moving objects in the scene, and associating sets of points in the keyframe with each object. Next we wish to estimate the rigid body motion of each object independently, using direct photometric alignment.

3.5 Estimating Transforms for Multiple Objects

Having identified each independently moving object, we wish to estimate a rigid body transform corresponding to that object's relative motion, as well as the motion of the camera itself. We may do so via a slight modification of the approach detailed in Section 3.2.

The pixels belonging to each active label from the segmentation $l \in \mathcal{L}$ correspond to disjoint subsets over all pixels in the keyframe, $\mathcal{X}_l \subseteq \boldsymbol{K}$.

We may then estimate a relative motion for each object by performing direct alignment on each subset independently, that is by restricting the computation of the residuals r_i to only those pixels in the subset, $\boldsymbol{x}_i \in \mathcal{X}_l$. The corresponding estimated transform, parametrized via Lie algebra is denoted $\hat{\boldsymbol{\xi}}_l$

The relative motion of the camera with respect to the world frame is found by performing direct alignment on the background set $\mathcal{B} \subseteq \boldsymbol{K}$, computed as the intersection of the inlier set $\mathcal{I}$ computed in Section 3.3, with the subset corresponding to the null label, $\mathcal{X}_\emptyset$, i.e $\mathcal{B} = \mathcal{I} \cap \mathcal{X}_\emptyset$. This corresponds to inlying pixels that do not belong to any moving object. The estimated transform is denoted $\hat{\boldsymbol{\xi}}_B$, and corresponds to the motion of the camera relative to the world.

Our approach has several key advantages. By identifying only moving objects, it is not necessary to compute a separate rigid body transform for every instance, as in [Bâr+18]. This reduces the model complexity and computational cost, and eliminates the need to rely on the semantic class to identify "movable" objects.

Additionally, by confining the relative motion estimate only to moving objects, more points are available to accurately estimate the ego-motion of the camera relative to the world.

We now have a method for computing points belonging to objects undergoing

independent rigid body motion, as well as a means of estimating their relative motion. However, the dense instance segmentation is often imperfect, typically including additional points that do not belong to the object corresponding to the instance (under-segmentation).

Points visible in the keyframe may also become occluded over a sequence of images, as objects move in front of one another, or behind elements of the world, thus changing the corresponding segmentations over time.

Additionally, points that do not belong to a given object but are associated to it through imperfect segmentation contribute to degrading the motion estimate from direct alignment, both by introducing points not belonging to the object into the motion estimate for that object, and by removing points from other objects.

We may repeat the process of outlier segmentation and motion estimation in order to jointly refine both the segmentation and the motion estimate. This is detailed in the next section.

3.6 Joint Segmentation-Transform Estimation

We have previously noted that the segmented set for a given label $\mathcal{X}_l$ is likely to be imperfect, and thus we wish to refine it by finding outlying points among this set, $\mathcal{O}_l \subseteq \mathcal{X}_l$.

For a given estimated transform $\hat{\boldsymbol{\xi}}_l$, we may compute an inlying set $\mathcal{I}_l = \mathcal{X}_l \setminus \mathcal{O}_l$ using graph cuts as in Section 3.3.2 by restricting the segmentation only to points $\boldsymbol{x}_l \in \mathcal{X}_l$. This refines the segmentation such that only points consistent with the estimated motion are marked as inliers.

However, we also noted previously that the presence of outliers among points used to perform the motion estimate will degrade the quality of the estimate. Thus we wish to jointly refine both the segmentation and the motion estimate to arrive at the best estimate of both. We therefore need a principled way of arriving at a jointly maximally likely transform and segmentation.

In order to include the influence of the binary inlier/outlier segmentation labeling $\boldsymbol{Y}_l \in \mathcal{Y}_l$ over $\mathcal{X}_l$, we construct the joint conditional distribution $p(\hat{\boldsymbol{\xi}}_l, \boldsymbol{Y}_l | \boldsymbol{R}_l)$. We may then restate the maximum posterior likelihood estimate of the transform as;

$$\hat{\boldsymbol{\xi}}_{lMAP} = \arg\max_{\hat{\boldsymbol{\xi}}_l} p\left(\hat{\boldsymbol{\xi}}_l | \boldsymbol{R_l}\right) = \arg\max_{\hat{\boldsymbol{\xi}}_l} \sum_{\boldsymbol{Y}_l}^{\mathcal{Y}_l} p\left(\hat{\boldsymbol{\xi}}_l, \boldsymbol{Y}_l | \boldsymbol{R}_l\right) \tag{3.53}$$

Finding $\hat{\boldsymbol{\xi}}_{lMAP}$ by exhaustively evaluating the sum over all possible labellings is generally computationally intractable even for a small set of points. For the same reason, we may not apply the standard Expectation-Maximization algorithm [Bis06] as computing an expectation over the labellings will not admit a closed-form solution, and requires exhaustive evaluation of all $2^{|\mathcal{X}_l|}$ possible labelings.

One possible approach is to use a mean-field approximation to the conditional random field, which allows the computation of an approximate expectation over labellings in closed-form. This is the approach taken in [SB15]. However, mean-field approximations require initialization close to the true posterior labeling in order to converge well, can be computationally intensive, and we lose the property of globally optimal MAP segmentations for a given residual image / transform.

Instead, we follow the so-called 'hard' expectation maximization approach [PD11][SCR12], as frequently applied to K-means clustering . Hard expectation-maximization is useful in the case where it is reasonably easy to compute a maximum likelihood estimate of the posterior values of latent variables, but difficult to compute their expected value.

The key idea is that rather than updating a joint distribution of model parameters and latent variables as in the standard expectation-maximization algorithm, we directly assign the maximum posterior likelihood estimate of the latent variables.

Let $\boldsymbol{F}$ be the new image frame. We perform joint segmentation-transform estimation according to the following algorithm;

Where the function R computes the residual image given an image frame, a set of points in the keyframe, and a transform, and the function assign constructs the inlier set given the labeling $\boldsymbol{Y}_l$. This algorithm is illustrated in figure 3.12.

We note that segmentation updates are computed over the full set $\mathcal{X}_l$, and not just the inliers at each step, while the transform is estimated only from the inliers. Additionally, we initialize each subsequent direct alignment step with the previous estimate of $\hat{\boldsymbol{\xi}}_l$.

Convergence occurs when either the change in the inlier set is sufficiently small, or when the magnitude of the residual vector $\boldsymbol{r}$ after direct alignment no longer decreases sufficiently between steps.

Algorithm 1 Joint Segmentation-Transform Estimation

$\mathcal{I}_l \leftarrow \mathcal{X}_l$
$\boldsymbol{R}_l \leftarrow R(\boldsymbol{F}, \mathcal{X}_l, \hat{\boldsymbol{\xi}}_I)$
while not converged **do**
  $\hat{\boldsymbol{\xi}}_l \leftarrow \arg\max_{\hat{\boldsymbol{\xi}}_l} p\left(\hat{\boldsymbol{\xi}}_l | \boldsymbol{R}_l; \mathcal{I}_l\right)$
  $\boldsymbol{R}_l \leftarrow R(\boldsymbol{F}, \mathcal{X}_l, \hat{\boldsymbol{\xi}}_l)$
  $\boldsymbol{Y}_l \leftarrow \arg\max_{\boldsymbol{Y}_l} p\left(\boldsymbol{Y}_l | \boldsymbol{R}_l\right)$
  $\mathcal{I}_l \leftarrow \text{assign}\left(\boldsymbol{Y}_l\right)$
end while

It can be shown that hard E-M is equivalent to coordinate descent optimization, and suffers from similar limitations; notably that it may become stuck at a non-stationary point in the case that the energy is not smooth, or else oscillate rather than converging. We will discuss this possible limitation in the discussion of experimental results.

Having an approach for jointly estimating rigid body transforms and a consistent segmentation, we now proceed to track multiple independently moving objects using photometric alignment, along with the camera motion. In the next section we discuss the complete algorithm to do so.

3.7 Complete Algorithm

We have now derived all the necessary parts for the full photometric odometry algorithm for dynamic objects. First, we describe the algorithm for estimating the relative transforms of multiple moving objects between a keyframe and a single new image. We then extend this to the case of a single keyframe and multiple images, before discussing the case of multiple updated keyframes.

3.7.1 Tracking objects in a single new image against a keyframe

Let the keyframe $\boldsymbol{K}$ consist of a set of pixels with corresponding intensities, and depths. Let $\boldsymbol{F}$ be a new image frame consisting only of pixels with corresponding intensities.

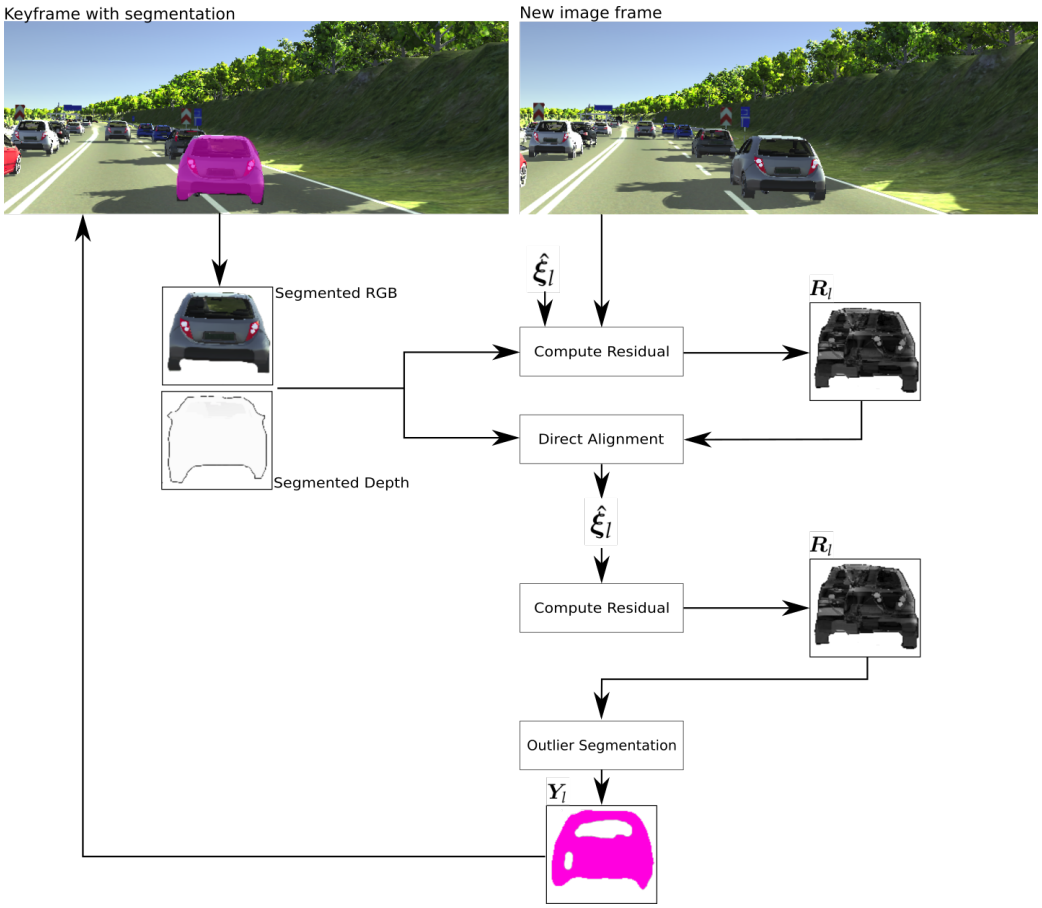


Figure 3.12: Illustration of the joint segmentation-transform optimization. Direct alignment is performed, resulting in a new residual image, which is then used to compute a new inlier/outlier segmentation. Note that initially, $\hat{\boldsymbol{\xi}}_l = \hat{\boldsymbol{\xi}}_l$, i.e. is initially set to the identity transform. The $\boldsymbol{Y}_l$ is set from the instance segmentation as in Section 3.4.

We first perform direct alignment as in Section 3.2 for all pixels in the keyframe. This gives an estimated transform $\hat{\boldsymbol{\xi}}_{\text{MAP}}$, which may be thought of as an estimate of the relative motion of the camera to the world under the assumption of a static scene.

We next compute the resulting residual image $\boldsymbol{R}$ over $\boldsymbol{K}$. We then use the residual

image to compute the disjoint inlying and outlying pixels via graph cuts, as in Section 3.3, producing the inlier set $\mathcal{I} \subseteq \boldsymbol{K}$, and outlier set $\mathcal{O} \subseteq \boldsymbol{K}$.

The keyframe pixels are segmented into disjoint, labeled sets using a neural network. We then determine which labels correspond to dynamic objects using the outlier set as in Section 3.4, giving rise to the active label set $l_1, l_2, \cdots, l_n \in \mathcal{L}$, each label corresponding to a set of pixels $\mathcal{X}_l \subseteq \boldsymbol{K}$.

In addition, we compute the set of pixels corresponding to the background $\mathcal{B}$ as those pixels in $\mathcal{I}$ which are not assigned an instance label by the neural network.

For each label $l \in \mathcal{L}$, we perform joint segmentation-transform estimation over the corresponding subset $\mathcal{X}_l$ against the new image frame, as detailed in Section 3.6. This produces estimates of the the transform $\hat{\boldsymbol{\xi}}_l$ corresponding to the relative motion of the camera to each dynamic object, as well as a refined outlier labeling for each object $\boldsymbol{Y}_l$.

We also perform joint segmentation-transform over the background set $\mathcal{B}$, which produces an estimate of the transform $\hat{\boldsymbol{\xi}}_{\text{CAM}}$ corresponding to the relative motion of the camera to the world frame.

We may now recover the transform corresponding to the motion of each object relative to the world frame, as in Section 3.1.2. Let C_{K} be the coordinate frame of the camera observing the keyframe, and let C_{F} be the coordinate frame of the camera observing the new image frame.

We define the origin of the world frame as coincident with the origin of the camera observing the keyframe, i.e. $\boldsymbol{T}_{WC_K} = \boldsymbol{I}$. Let O^l be the coordinate frame attached to the object with label l.

Following equation 3.18, we have $\boldsymbol{T}_{WC_{\text{F}}} = \exp(\hat{\boldsymbol{\xi}}_{\text{CAM}})$ and $\boldsymbol{T}^{O^l}_{C_F C_K} = \exp(\hat{\boldsymbol{\xi}}_l)$. We may attach the coordinate frame O^l as desired; we choose this as the center of the object, initially aligned with the world coordinate frame, and construct $\boldsymbol{T}_{C_K O^l} = \boldsymbol{T}_{WO^l_K}$ appropriately.

In the absence of a given center for each object, we estimate a centroid using the average over the depths of the inlying points after segmentation refinement. Thus we have;

$$\begin{aligned} \boldsymbol{T}_{WO_F^l} &= \boldsymbol{T}_{WC_F} \boldsymbol{T}_{C_F C_K}^{O^l} \boldsymbol{T}_{C_K O^l} \\ &= \exp(\hat{\boldsymbol{\xi}}_{\text{CAM}}) \exp(\hat{\boldsymbol{\xi}}_l) \boldsymbol{T}_{C_K O^l} \end{aligned} \tag{3.54}$$

Which is the transform corresponding to the motion of each object in the world frame. We now extend the algorithm to the case of a single keyframe and a sequence of new image frames.

3.7.2 Tracking objects over multiple new image frames against a keyframe

Extending the algorithm to handle tracking multiple objects over a sequence of new image frames $\boldsymbol{F}_1, \boldsymbol{F}_2, \cdots \boldsymbol{F}_k$ is straightforward. Intuitively, we use the results of the estimates from frame $\boldsymbol{F}_{k-1}$ to initialize the algorithm for the next frame $\boldsymbol{F}_k$.

We apply the following modifications to the algorithm as presented in the previous section.

We initialize the first "static scene" direct alignment step for frame k as $\boldsymbol{\xi}_{I_k} = \boldsymbol{\xi}_{\text{CAM}_{k-1}}$. This encodes the assumption that the motion of the camera with respect to the world is relatively small between consecutive new image frames, and thus such an initialization will be close to the true estimate.

We perform the full-frame outlier segmentation and label activation steps as before, noting that objects which were not yet moving sufficiently in the first frame may begin moving in subsequent frames. This gives the active set for each image frame, $\mathcal{L}_k$ as well as the background set $\mathcal{B}_k$.

For each label $l \in \mathcal{L}_k$, we perform joint segmentation-transform estimation as before. If we have performed joint segementation-transform estimation for the label l in the previous frame, we initialize with the previous estimate of the transform $\hat{\boldsymbol{\xi}}_{l_{k-1}}$, and the previous estimate of the inliers $\mathcal{I}_{l_{k-1}}$. This gives the estimate of the transform corresponding to the relative motion of the object from $\boldsymbol{K}$ to $\boldsymbol{F}_k$ as $\hat{\boldsymbol{\xi}}_{l_k}$

Note that in each iteration of the joint segmentation-transform estimation, we perform segmentation over the full set of labeled pixels $\mathcal{X}_l$. This allows us to account for objects being partially occluded and decluded as they move, with respect to their visibility in the keyframe.

We perform joint segmentation-transform estimation to arrive at an estimate of the transform corresponding to the camera motion, using the set $\mathcal{B}_k$, giving $\hat{\boldsymbol{\xi}}_{\text{CAM}_k}$.

We may then estimate the transform corresponding to the motion of each object. Attaching the coordinate frame O^l as in the previous section, we compute the transform from object to world coordinate frame for new image frame $\boldsymbol{F}_k$ analogously to equation 3.54;

$$\boldsymbol{T}_{WO_k^l} = \exp(\hat{\boldsymbol{\xi}}_{\text{CAM}_k}) \exp(\hat{\boldsymbol{\xi}}_{l_k}) \boldsymbol{T}_{C_K O^l} \tag{3.55}$$

We may now compute a sequence of transforms $\boldsymbol{T}_{WO_1^l}, \boldsymbol{T}_{WO_2^l}, \cdots \boldsymbol{T}_{WO_k^l}$ for each object, which correspond to the trajectory of a given object over time. Additionally, we have a sequence of transforms $\boldsymbol{T}_{WC_1}, \boldsymbol{T}_{WC_2}, \cdots \boldsymbol{T}_{WC_k}$ corresponding to the trajectory of the camera. Thus we may track both the motion of the camera, and the motion of objects over a sequence of frames.

It is not feasible to track against a single keyframe indefinitely. The motion of the camera may result in fewer points from the keyframe being visible in subsequent image frames. Additionally, dynamic objects may enter and leave the camera's field of view, requiring a new instance segmentation to be generated.

We therefore turn to the problem of tracking with multiple keyframes in the next section.

3.7.3 Tracking with multiple keyframes

Tracking with multiple keyframes requires further modifications to the algorithm. The coordinate frame of subsequent keyframes is no longer necessarily coincident with the origin of the world frame. Thus we must also account for the relative pose of subsequent keyframes.

Additionally, because the instance segmentation for each keyframe is generated independently, instance labels in one keyframe may not correspond to the same label in another. We must therefore associate labels from one keyframe to another so that we may continue to track the same object across frames.

We assume that a keyframe is constructed from an image frame by adding depth and segmentation information. Let the coordinate frame associated with the initial keyframe $\boldsymbol{K}_0$ be coincident with the world frame. Then if a new keyframe is constructed from

some new image frame $\boldsymbol{F}_k$, the relative pose of the new keyframe $\boldsymbol{K}_1$ is given by the estimate of the camera pose with respect to the initial frame, i.e. $\boldsymbol{T}_{K_0K_1} = \exp(\hat{\boldsymbol{\xi}}_{\text{CAM}_F})$.

We treat the coordinate frame attached to the new keyframe as the origin for subsequent tracking, i.e. the transform corresponding to the camera motion $\exp(\hat{\boldsymbol{\xi}}^j_{\text{CAM}})$ and to the motion of each object with label$l$, $\exp(\hat{\boldsymbol{\xi}}^j_l)$, are expressed with respect to the coordinate frame corresponding to the most recent keyframe $\boldsymbol{K}_j$.

A pose with corresponding transform $\boldsymbol{T}^J$ in the coordinate frame of keyframe $\boldsymbol{K}_j$ can be expressed in the world frame as $\boldsymbol{T}^W = \boldsymbol{T}_{K_0K_1}\boldsymbol{T}_{K_0K_1}\cdots\boldsymbol{T}_{K_{j-1}K_J}\boldsymbol{T}^j$. Thus we may recover the trajectory of the camera and of each dynamic object in the world frame.

We associate an instance label $l \in \mathcal{L}_i$ in the active set of keyframe $\boldsymbol{K}_i$ with an instance label $m \in \mathcal{M}$ in the instance segmentation of $\boldsymbol{K}_j$ as follows.

Assume we have an estimated transform corresponding to the object motion observed in $\boldsymbol{K}_i$ to the image frame of $\boldsymbol{K}_j$, expressed as $\boldsymbol{T}^i_{l_k}$. We apply the warping function from equation 3.21 to each pixel in $\boldsymbol{x}^i_l \in \mathcal{X}^i_l$ which are labeled inliers, to determine the corresponding pixel in the image frame of $\boldsymbol{K}_j$, $\boldsymbol{x}^j_l$, i.e. $\boldsymbol{x}^j_l = \tau\left(\boldsymbol{x}^i_l, d, \boldsymbol{T}^i_{l_k}\right)$. We refer to the set of corresponding points as $\mathcal{X}^j_l$.

We compute the intersection-over-union of $\mathcal{X}^j_l$ with each set of pixels $\mathcal{X}_m$ corresponding to each label $m \in \mathcal{M}$. The label l is then associated with the label m if the intersection-over-union score is higher than a chosen threshold, and is also the highest among all labels in $\mathcal{L}_i$. Note that this means at most one label in $\mathcal{L}_i$ can be associated with each label in $\mathcal{M}$, and some labels in $\mathcal{L}_i$ may not be associated at all. We assume that tracking of unassociated labels is lost.

We now have a mapping between sets of labels in keyframes. Thus we can track the same object over multiple keyframes, and recover its trajectory in the world frame. We are also able to recover the camera trajectory from multiple keyframes. Thus we have completed the description of the dynamic photometric odometry algorithm.

In the next section, we discuss implementation details of the algorithm into account, and evaluate its effectiveness in the task of tracking dynamic objects.

4 Evaluation

In this chapter we detail the evaluation of various aspects of the complete dynamic photometric odometry system, in order to determine its performance and limitations. We evaluate the complete system against datasets providing ground-truth for dynamic object motion, as well as camera motion through the scene.

We focus in particular on the system's effectiveness in tracking dynamic objects, as well as in refining an initial noisy instance segmentation.

4.1 Implementation Details

The full system is implemented in the MATLAB mathematical programming environment, primarily for ease of debugging and prototyping. Evaluations are run on a commodity laptop with an Intel Core i7-4720HQ CPU running at 2.60 GHz, with 16GB of RAM.

We utilize a MATLAB implemetation of the Boykov-Kolmogorov Maxflow algorithm [Ant14], which includes a simple interface for the construction and optimization of graph-cut problems.

4.2 Oxford Multi-Motion Dataset

The Oxford Multimotion Dataset (OMMD) [JG19] is a recent, comprehensive dataset from the Oxford Estimation, Search, and Planning Research Group.

It includes stereo video sequences of scenes involving multiple independently moving rigid bodies with combinations of rotational and translational motions, as well as ground truth motions for these objects as measured by a high-speed Vicon motion capture system.

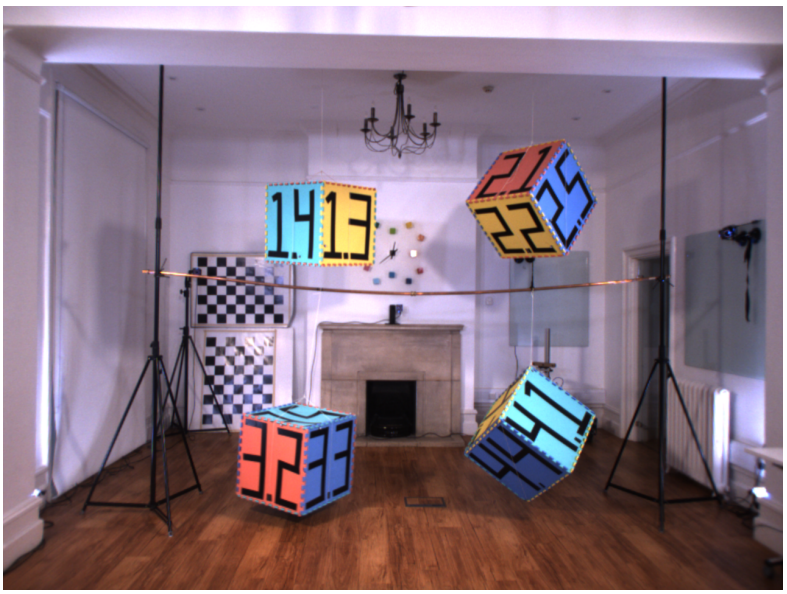


Figure 4.1: Example image frame from the Oxford Multimotion Dataset. This example is drawn from the left camera of the stereo rig, in the *swinging-static* sequence. Note that boxes are labeled according to their number and visible face, i.e. face 1.4 is the face labeled 4 of box 1.

4.2.1 Data

The dataset is made up of several synchronized, rectified stereo video sequences recorded with a Point Grey Bumblebee XB3 stereo camera at 16 Hz and 1280x960 resolution. Additionally, the dataset contains corresponding RGB-D sequences recorded via an Intel Realsense D435 system at 30 Hz and 640x480 resolution, attached to the same apparatus. An example frame from the left camera of the stereo rig is shown in figure 4.1.

The recording apparatus also includes a rigidly attached Microstrain 3DM-GX4-45 IMU, recording orientation and acceleration data at 500 Hz, as well as a marker allowing for tracking by the Vicon motion capture system at 200 Hz. Each moving object is assigned a coordinate frame in order to record the ground-truth motions, also at 200 Hz.

Intrinsic camera calibrations, as well as extrinsic calibrations between each sensor frame are supplied as part of the dataset.

The three *swinging* sequences from the dataset are of particular interest for our evaluation. These contain scenes involving four coloured and numbered boxes undergoing distinct, constrained motions - swinging side to side, back and forth, as well as rotating about an axis and precessing. We are therefore able to evaluate the effectiveness of the system in tracking each type of motion in isolation.

The scenes are recorded from a stationary apparatus in the *swinging-static* sequences, from an apparatus undergoing only translational motion in the *swinging-translational* sequence, and with the apparatus moving in an unconstrained way in the *swinging-unconstrained* sequence.

An important property of the motions of the boxes is their periodicity. The periodic motions of the objects permit the observation of the same motion over many repetitions. Using the ground truth trajectories provided in the dataset, the translational period of boxes 1 and 3 was observed to be approximately 40 frames, whereas the rotational period of boxes 2 and 3 is approximately 125 frames.

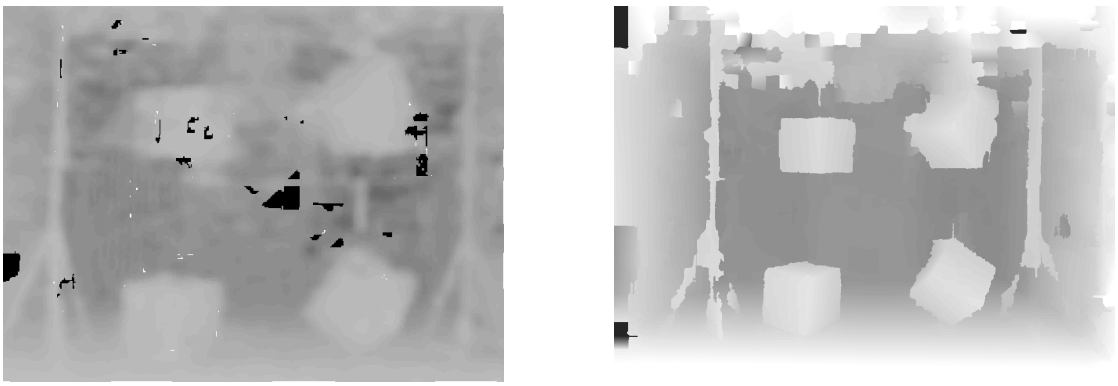
These periods do not vary significantly over the first several hundred frames of the dataset.

4.2.2 Preprocessing

Our system depends on reasonably good depth estimates. We found that the depth recorded from the Intel Realsense sensor was too noisy, with many artifacts producing warped estimates for the 3D structure of the moving boxes. Additionally, it was of low resolution in comparison to the images from the stereo cameras. We therefore computed a new depth estimate through stereo matching.

Because the scene and objects of interest consist mainly of planar surfaces in various orientations, we chose to use Slanted Plane Smoothing Stereo (SPS-Stereo) [YMU14]. SPS-Stereo was used to compute a pixel-wise disparity estimate from the left stereo frame to the right.

We then compute the depth d from disparity D for each pixel as $d = \frac{fB}{D}$, where B is the stereo baseline computed from the extrinsic calibration parameters provided by the dataset, and f is the focal length from the provided intrinsic calibration for the left camera. The resulting depth estimate contained fewer artifacts and was found to be suitable for our evaluation. Example outputs of this process compared to the raw Intel Realsense output are show in figure 4.2.



Depth image from Intel Realsense. Disparity image from SPS-Stereo.

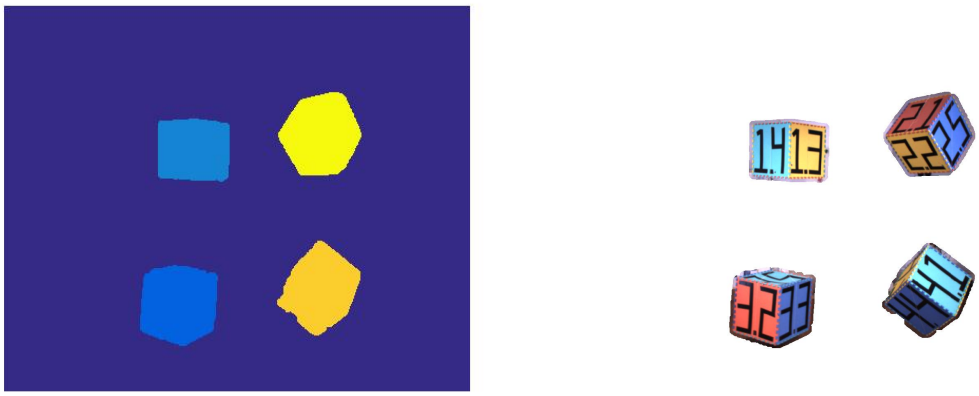
Figure 4.2: Comparison of Intel Realsesne (left) and SPS-stereo output (right) on the same example frame. Note the presence of many artifacts where depth is not known (black), as well as the overall lack of definition in the raw Intel Realsense data. In contrast the SPS-stereo disparity image has far fewer artifacts and significantly improved definition, particularly at the objects of interest.

An additional preprocessing step is required to find an initial segmentation for the objects of interest in the OMMD. Most off-the-shelf instance segmentation neural networks are unsuitable for segmenting the moving boxes, as they do not belong to any class that such networks are trained for.

Training a network architecture to segment these objects would be time consuming, and require labeled training data which is not available. We therefore took the simple approach of color segmentation in HSV colourspace [AA05].

We note that the sides of the boxes are brightly coloured and distinct from the background. We therefore partition the HSV colourspace into object and background by manually finding appropriate thresholds. Pixels falling into the object partition are assigned to connected regions.

Regions that have fewer than 1% of all pixels in the image are eliminated. The remaining regions are dilated, and finally assigned an instance number. The output of this process is illustrated in figure 4.3.



Instance Segmenation. Instance Mask.

Figure 4.3: Instance segmentation, colored by instance label (left), with the corresponding masked regions in the image (right).

We note that this approach results in an under-segmentation, i.e. instance segments include all pixels belonging to the object of interest, but also some pixels from the background. It is often the case that neural networks also provide imperfect segmentations, and we will evaluate whether the joint segmentation-transform estimation in section 3.6 can improve upon the initial segmentation.

Having arrived at improved depth estimates and initial segmentations, we can begin our evaluations.

4.2.3 Static Camera

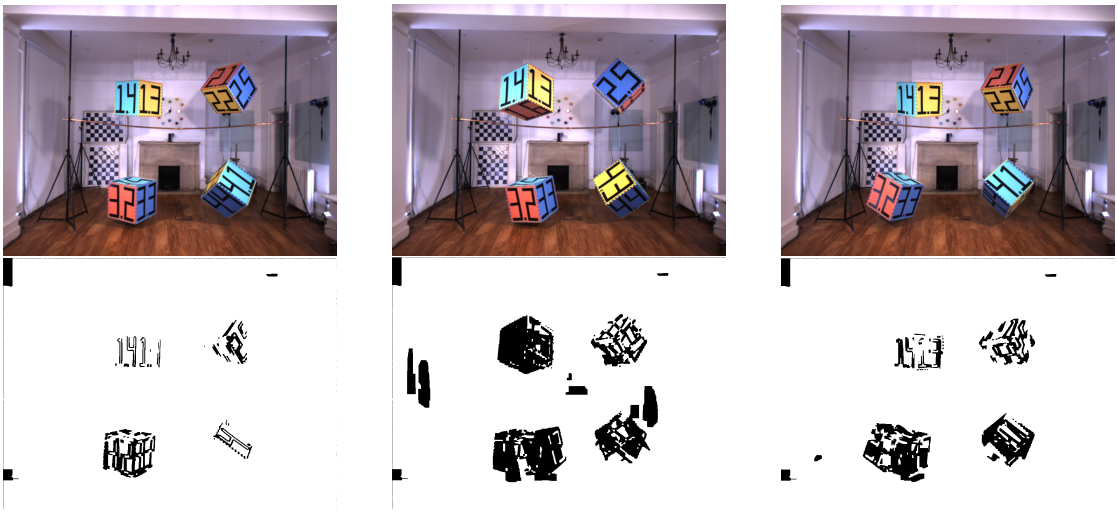
The *swinging-static* sequence involving a static camera apparatus allows us to evaluate the effectiveness of our system for the tracking of moving objects independently of the motion of the camera.

Segment Activation

In order to determine an appropriate threshold for activating instance labels for tracking via motion segmentation as in 3.4, we observe the fraction of points belonging to a

given instance that are segmented to outlier.

An illustration of how the outlier segmentation changes over the course of several frames is shown in figure 4.4.



Frame 1 Frame 50 Frame 120

Figure 4.4: Outlier segmentation over time. Black pixels correspond to those segmented to outlier. Note that as well as outliers due to the motion of boxes in the scene, there are additional outliers due to shadows caused by box 1 swinging in front of one of the light sources. This varies the illumination for sections of the background, resulting in high photometric residuals even though the camera is static. Note also additional outliers caused by poor depth estimates (e.g. top left corner).

We take the first frame in the sequence (frame 0) as the keyframe, and compute the residual segmentation over 150 frames, without estimating any transforms. The proportions are plotted in figure 4.5. We initially choose $\sigma = 0.1$ and $w = 1.0$ as the unary and pairwise parameters for the segmentation energy in section 3.3.3.

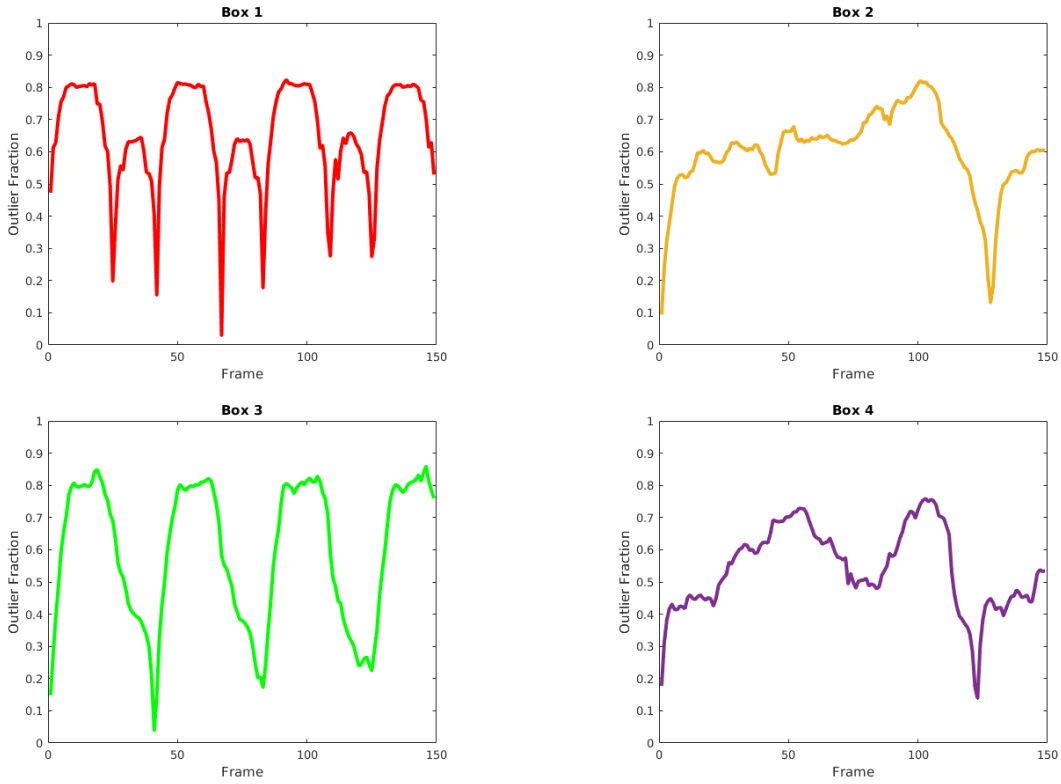


Figure 4.5: Fraction of pixels belonging to each instance segmented to outlier for each frame, without transform estimation.

We note the strong periodicity apparent in the plots, which conforms to the observed periodicity of the motion of the objects. Residuals are small when the pose of the object is close to where it is observed in the keyframe, and increase as the pose diverges.

In the case of box 1, which is swinging toward and away from the camera, there are two times when the residual is small in a given period. This is because the initial pose does not lie at the end of the arc, but at approximately 1/3 from the top of the nearest

point in the arc.

We note that as the object pose diverges over time, the pixels corresponding to each instance segment that are segmented to outlier increases rapidly, and remains stably high until the object returns close to the initial pose. This suggests that a relatively high activation threshold can be used; we chose 0.3 for subsequent experiments, i.e. 30% of the pixels belonging to a given instance must be segmented to outlier, in order to activate that segment for tracking.

We also observed the fraction of pixels that do not belong to any object instance that are segmented to outlier, plotted in figure 4.6.

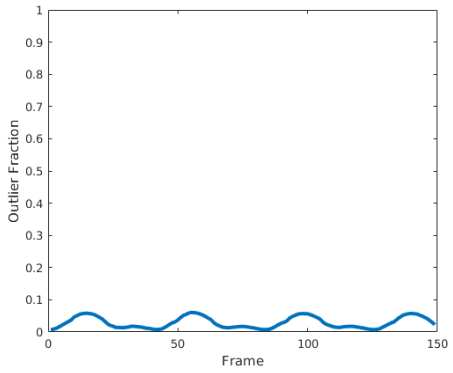


Figure 4.6: Fraction of outlier pixels not belonging to any instance.

The outliers that are not assigned to any instance (unlabeled) are due to moving objects occluding the background, as well as illumination changes caused by moving objects occluding light sources.

We note also that periodicity is present for the background, supporting that outliers are largely caused directly or indirectly by moving elements of the scene.

These results suggest that significant information about dynamic scenes can be extracted by computing and evaluating the photometric residual.

We now turn to the use of direct photometric alignment for the tracking of dynamic objects.

Object Tracking With a Static Camera

We proceed to evaluate the effectiveness of direct photometric alignment as an approach to tracking dynamic objects. We observe tracking effectiveness with both the *naive* direct alignment aglorithm as presented in section 3.2, including robustification via Huber weights and coarse to fine optimization. For comparison we then evaluate the effectiveness of the joint segmentation-motion estimation as in the full system presented in section 3.7.

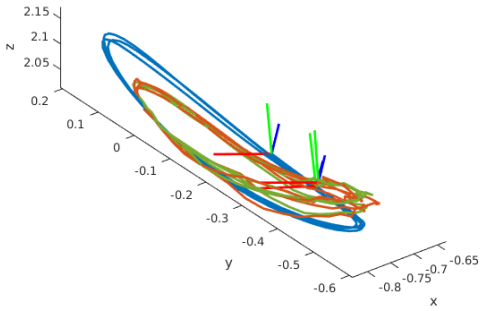
In order to compare the estimated trajectories with the ground truth provided by the vicon system, we apply the extrinsic calibrations provided with the dataset to transform the estimates to the vicon frame.

Additionally, we note that the coordinate frame attached to the centroid estimate (section 3.7.1) does not correspond to the coordinate frame attached to the object in the vicon frame. In order to compensate for this, we compute $\mathbf{T}^{W}_{\text{cent gt}}$ the transform from the ground truth pose of the object in the first frame in the image sequence, to the centroid pose in the first frame in the sequence, in the world coordinate frame.

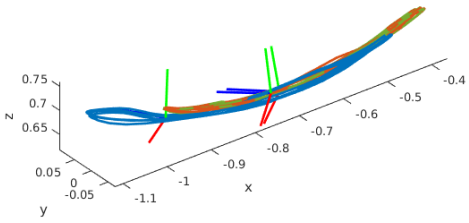
This allows us to transform the estimated trajectory such that it applies to the coordinate frame attached to the object in the vicon system. We evaluate the estimated trajectories accordingly.

We first evaluate the tracking of the objects undergoing only translational motion, boxes 1 and 3. Since the camera is stationary, and the same faces of the boxes are visible throughout the sequence, we use only the first image frame, and the associated depth estimate and instance segmentation, as the keyframe.

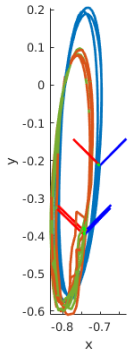
The estimated tracks are plotted in figure 4.7. The trajectory plots reflect the poses of the objects in the Vicon frame, $\mathbf{T}_{\text{Vicon obj}}$ in our convention. Qualitatively, we observe that the shape of the estimated trajectory is broadly consistent with that of the ground truth. Several features stand out.



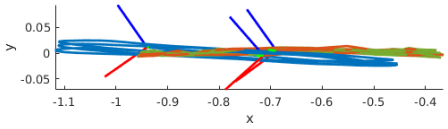
Box 1, 3D Plot



Box 3, 3D Plot



Box 1, Projection to XY plane.



Box 3, Projection to XY plane.

Figure 4.7: Ground truth (blue), naive estimates (green), and joint segmentation-motion estimated tracks (orange) for box 1 and 3. Tracking is performed against a single keyframe, from the start of the image sequence. All plots show the pose of the object in the the vicon coordinate frame. The plotted coordinate axes show the final pose of each trajectory, to illustrate the total drift. Note the over and undershoot in the y axis for Box 1, and the x axis for Box 2.

The trajectory estimates appear to overshoot the ground truth in one direction, and undershoot in the other. In the case of Box 1 (left side of figure 4.7), the overshoot is in the direction of the camera ($-y$), and the undershoot is away from it. In contrast, the estimate for Box 3 (right side of figure 4.7) tends to overshoot toward the right ($+x$) and undershoot toward the left.

The overshoot in both cases is in the direction of the initial motion. A possible explanation is that an initial overshoot may shift the trajectory in that direction, resulting in the estimate of the periodic trajectory being biased that way.

An alternative explanation may be that the initial coordinate frame is coincident, but not aligned. Since we track against the first frame, and do not perform additional alignment between estimated and ground truth trajectories, an initial misalignment would propagate through the trajectory estimate, and may introduce a constant bias bias in the poses, as we have observed.

We note also in the case of Box 1 that the estimated trajectory is considerably less smooth closer to the camera than away from it.

It might be the case that the relative motion in image space is larger closer to the camera, and hence the initialization of the direct alignment step from the previous frame's transform estimate is further from the true minimum, resulting in the algorithm finding a closer, inferior minimum.

We compute the absolute trajectory error (ATE) between the ground truth and the estimated trajectories, as in [Stu+12]. Results are tabulated in 4.1.

Estimate	**Mean ATE** (m)	**Max. ATE** (m)	**RMSE** (m)
Box 1, Naive	0.1112	**0.2213**	0.1276
Box 1, Full System	**0.1101**	0.2245	**0.1267**
Box 3, Naive	**0.1258**	0.2565	0.1501
Box 3, Full System	0.1261	**0.2556**	**0.1500**

Table 4.1: Mean ATE, Maximum ATE, and RMSE between the ground truth and estimated trajectories. The best results for each box are bolded.

The absolute trajectory error is an expression of the absolute translational distance between the ground truth and estimated trajectories, for each frame. We state the mean, maximum, and root mean square (RMSE) values for the ATE for each estimate.

Note that because we have transformed our estimated trajectories into the Vicon coordinate frame, we do not perform additional alignment between estimate and ground truth when stating our results.

Quantitatively, the lack of consistent improvement due to the joint segmentation-motion step in the full system, over the naive direct alignment algorithm, is somewhat surprising.

Several factors may contribute. One possibility is that the initial segmentation is already relatively good, and thus true outliers represent only a small fraction of the points assigned to an instance. Thus there may be little improvement to be had by further refining the segmentation through eliminating outliers.

Another possibility is that the Huber weighting of the residual in the direct alignment step already reduces the influence of outlying residuals sufficiently well, making the segmentation step unnecessary.

A third possibility is that along with eliminating outliers, the spatial pairwise term in the outlier segmentation energy results in inlying points being erroneously segmented to outlier. This causes information to be lost, for a net result of no improvement in the tracking.

Finally, it is possible that misaligned initial frames may introduce a bias, giving rise to errors which overshadow any improvements due to joint segmentation-motion optimization.

We next attempt to estimate the trajectories of the rotating objects, i.e. boxes 2 and 4.

Due to the rotation of the objects, the initially visible faces are no longer visible after approximately 30 frames. We therefore choose to construct a new keyframe every 15 frames.

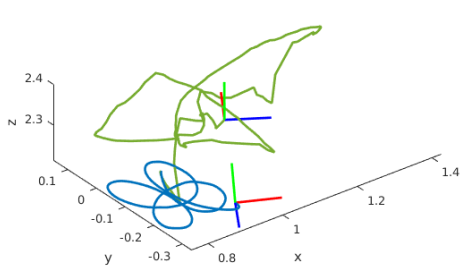
We note that the estimate of the centroid varies from keyframe to keyframe. In order to account for this, we transform each pose according to the relative transform from the centroid associated with the current keyframe, to that associated with the previous keyframe. This results in a trajectory estimate that is consistent with the initial centroid estimate.

The system proved to be incapable of estimating the rotational motion of these objects accurately. The estimated trajectory diverges quickly and completely from the ground truth for both rotating objects.

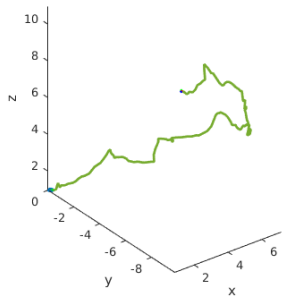
Additionally, in both cases, joint segmentation-motion estimation causes over 90% of the pixels belonging to the instance segmentation to be marked as outlier over approximately 4 frames, and subsequently tracking is lost. The estimated trajectories are plotted in figure 4.8.

The failure to track rotating objects represents a very significant limitation of the

system. This result is also surprising, given that direct photometric alignment is known to successfully estimate camera motion under both rotation and translation [KSC13][ESC14][EKC18].



Box 2, 3D Plot



Box 4, 3D Plot

Figure 4.8: Ground truth (blue), naive estimates (green) for box 2 and 4. All plots are in the world coordinate frame. Note that the ground truth for Box 4 is barely visible in the plot, as the divergence is very large. Joint segmentation-motion estimates proved to be impossible, as a very large fraction of the instance was quickly segmented to outlier.

The trajectory estimate diverges from the ground truth quickly, over only a few frames. Addtionally, a large fraction of the pixels belonging to the object instance label are segmented as outliers, as that the photometric residual increases quickly.

This suggests the failure to track rotating objects is due to the underlying direct photometric alignment algorithm. This is somewhat surprising and merits further investigation.

As discussed in section 3.1.2, observations of the motion of an object from a static camera are equivalent to observations from a camera moving relative to a static object.

In the case of a rotating object, the corresponding *relative* camera motion is an arc with radius equal to the distance between the object's center of rotation and the camera center, lying in the plane defined by the axis of rotation. This is illustrated in figure 4.9.

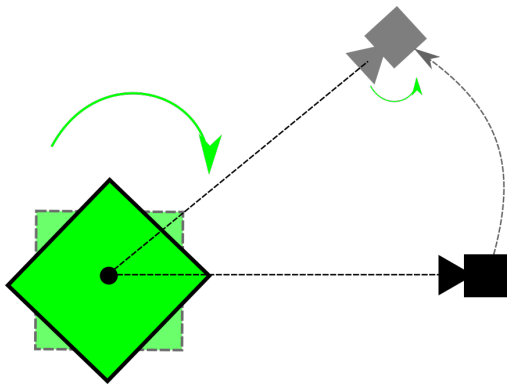


Figure 4.9: An object (green) is rotating clockwise in space. The relative motion of the camera (black) has a rotational and a translational component, in the opposite direction to the rotation of the object. The translational component increases as the radius of the arc increases. The rotational component are equal and opposite for camera and object.

Thus rotational motion in the object frame introduces a potentially large relative translational motion. This large relative translational motion may place the minimum found by the underlying Gauss-Newton algorithm too far from the initialization, and hence may result in poor convergence. This conforms with the observation that photometric odometry also gives good estimates for rotating cameras; the camera rotation is about the camera origin, and hence does not give rise to an additional translational component.

Direct photometric alignment may handle large translations through an increase in the number of pyramid levels in the coarse to fine optimization, as discussed in section 3.2.4. However, as we noted therein, it becomes quickly impossible to use additional pyramid levels for object tracking as the object becomes invisible.

An alternative approach may be to re formulate the warping function stated in equation 3.21. The stated formulation as used in deriving the required energy minimization takes as argument the relative motion of object and camera.

Instead, we propose to formulate the warping function such that it takes as argument the transform corresponding to the motion of the object in a frame of reference close to the object center, thus reducing the induced relative translation due to rotation. We leave the required derivation and evaluation to future work.

Another possibility is that these object motions result in a pathological case for the linearization applied to the non-linear least-squares problem in section 3.2.4.

In order to investigate this possibility, we implemented the Levenberg Marquardt (LM) algorithm [Mar63] for nonlinear least squares optimization. The LM algorithm augments the Gauss-Newton algorithm through the use of a damping term coupled to a gradient descent step.

If the Gauss-Newton step increases the error, the LM algorithm increases the weight associated with the gradient descent term, thus damping possible divergence.

Despite this, we did not see significant improvement, and the trajectory estimate diverged in the same manner. The stability of the linearization and gradient terms for observations on rotating objects should also be a focus of future work.

One final possibility is that insufficient information is contained in the relatively few points belonging to the instance segment of a rotating object, to accurately estimate rotations. Initially available information is also quickly lost, as rotation causes self-occlusion of initially visible regions. This may be compounded through inaccurate depth estimates.

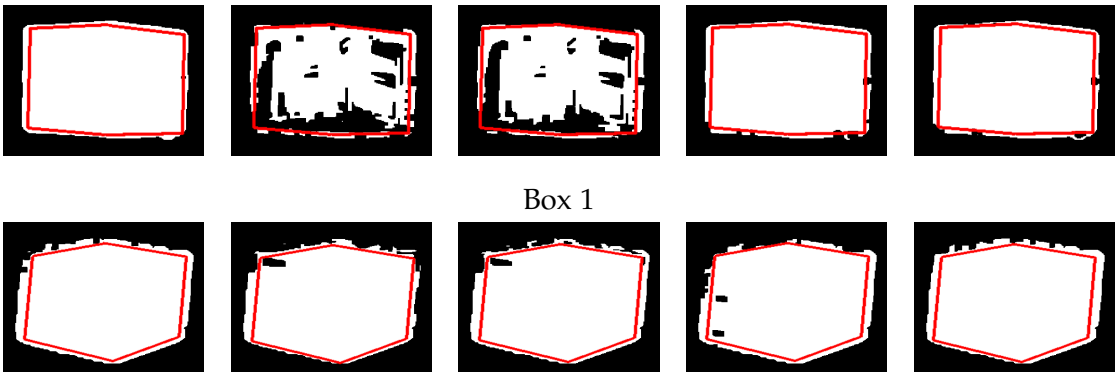
How much information, whether through number of pixels, their texture, or the accuracy of depth information, is required to track a dynamic object should also be examined in future work.

Having examined the possibility of tracking dynamic objects with direct photometric alignment, we now turn to the effectiveness of refining an initial segmentation using joint segmentation-motion estimation.

Segmentation refinement

In order to determine the effectiveness of using the photometric residual to refine an initial instance segmentation, we observe the results of outlier segmentation on the instance segments for the successfully tracked dynamic objects, boxes 1 and 3.

We perform outlier segmentation after each transform estimate of the naive direct photometric alignment algorithm, and compare it with the outlier segmentation from the joint segmentation-motion estimate. Example segmentations over a sample of the frame sequence is shown in figure 4.10.



Box 1

Box 3

Figure 4.10: Segmentation evolutions for Box 1 and Box 3, sampled at every 10th frame in the 50 frame sequence. Black pixels correspond to those segmented to outlier, or not belonging to the initial instance segmentation. The true boundary of the object is shown in red. This example is drawn from the joint segmentation-motion estimate, the naive estimate is similar.

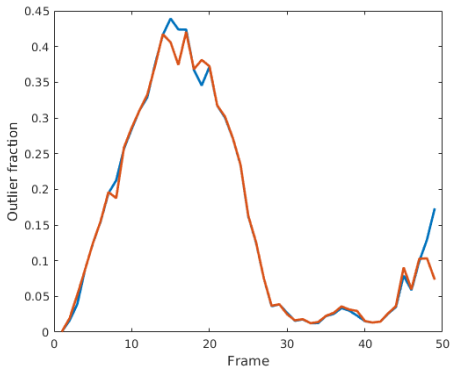
Qualitatively we observe that the segmentation refinement is relatively poor for Box 1, most likely due to the relatively poor motion estimates resulting in high photometric residuals in many areas of the object.

Background pixels remain segmented to inlier, most likely due to the lack of background texture in this part of the scene.

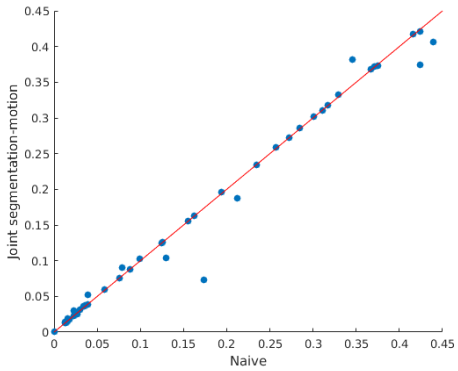
In contrast, the segmentation for Box 3 appears to correctly segment parts of the background to outlier, while generally retaining as inlier pixels belonging to the object. Additionally, small internal outlier segments emerge as specularity causes the irradiance of certain points of the objects to change.

Next we evaluate the dynamics of the outlier segmentation, comparing the outlier segmentation of the residual after naive direct photometric alignment, with that from the joint segmentation-motion estimate.

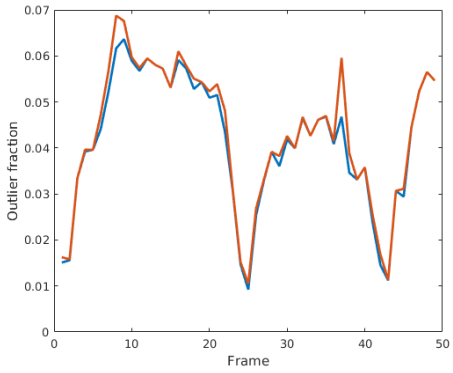
We observe the fraction of the number of pixels belonging to each instance that is segmented to outlier over a full period of the motion of each object. These results are plotted in figure 4.11.



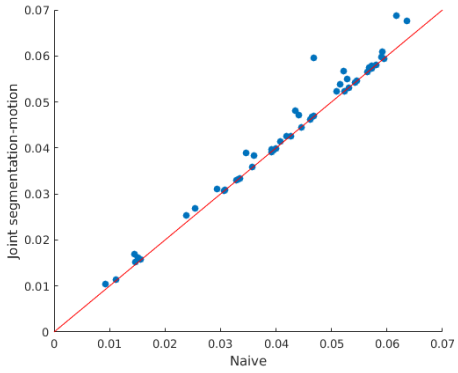
Box 1, outlier fraction per frame.



Box 1, Joint VS Naive outlier fractions.



Box 3, outlier fraction per frame.



Box 3, Joint VS Naive outlier fractions.

Figure 4.11: Fraction of pixels belonging to a given instance that are segmented to outlier from naive direct alignment (blue) and joint segmentation-motion estimation (orange) in each frame. The 45-degree line in the right-hand plots indicates equality.

Several features are immediately apparent. The first is the periodicity of the outlier fraction, which matches the periodicity of the object's motion. In the case that pixels not belonging to the objects were correctly segmented to outlier, we would expect a steady state in the outlier fraction corresponding to the proportion of 'true' outliers. However, this is not apparent here.

Additional sources of large photometric residuals leading to pixels being erroneously

segmented to outlier include illumination changes, such as increased irradiation as the boxes move closer to and further from a point light source, as well as large photometric residuals induced by imperfect motion estimates.

Additionaly, pixels not belonging to the object may not be segmented to outlier when the background texture is insufficient, as illustrated in section 3.4.

The lack of ground truth segmentation for the OMMD makes it infeasible to evaluate the contribution of these factors, which should be examined in future work. The use of synthetic scene data may be of particular importance.

We note that the joint estimate is very similar to the segmentation from the naive transform estimate, but with some large difference evident. We note that the differences tend to occur near extrema, i.e. the difference between the two segmentations is greatest at the maximal and minimal fractions.

The reason for this discrepancy is not clear, and open to further investigation.

Having evaluated the effectiveness of joint segmentation-motion estimation in refining an initial instance segmentation, we next turn to evaluating scenes with dynamic objects observed from a moving camera.

4.2.4 Dynamic Camera

The *swinging-translational* and *swinging-unconstrained* sequences involve moving the camera apparatus through the scene. In *swinging-translational* the camera undergoes only translational motion, while in *swinging-unconstrained* the camera both translates and rotates.

These scenes allow us to evaluate the feasibility of tracking dynamic objects from a dynamic camera. Additionally, we evaluate the accuracy of camera tracking through dynamic scenes.

Object Tracking with a Dynamic Camera

We apply the activation threshold ($t = 0.3$) computed experimentally in the previous section to activate instances for tracking.

We found that we were unable to track any of the four dynamic object, for either of the sequences involving a dynamic camera. The transform estimates diverge sharply, until tracking is lost entirely, with similar dynamics as the estimates for rotating objects

in the previous section.

The tracking divergence of even the translating objects, boxes 1 and 3, could be attributable to the change in relative bearing between the camera center and the center of each object. This change in relative bearing may be regarded as a rotational motion of the object relative to the camera.

Thus the cause of tracking failure is likely to be the same as for the rotating objects in the previous section, and may reflect some combination of the same causes. An important direction of future work is to focus on the feasibility of estimating the motions that include relative rotations through direct photometric alignment.

Camera Tracking in a Dynamic Scene

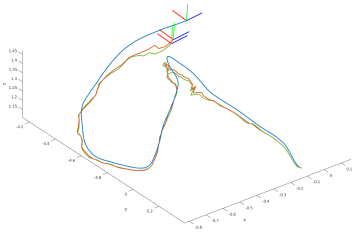
We evaluate the effectiveness of joint segmentation-motion estimation in estimating the trajectory of a camera moving through a dynamic scene, using both naive direct alignment and joint segmentation-motion estimation.

The estimated camera trajectories are plotted in figure 4.12. We also evaluate the absolute trajectory error (ATE) and the Root Mean Square Error (RMSE) between the ground truth and the estimated camera trajectories.

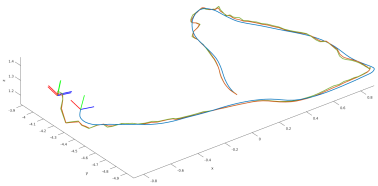
Note that since the ground truth and estimated trajectory are in the same coordinate frame, we do not perform Horn alignment. These results are tabulated in table 4.2.

Estimate	**Mean ATE** (m)	**Max. ATE** (m)	**RMSE** (m)
Translational, Naive	0.0572	0.1674	0.0661
Translational, Full System	**0.0528**	**0.1406**	**0.0592**
Unconstrained, Naive	0.0494	0.2137	0.0594
Unconstrained, Full System	**0.0474**	**0.2026**	**0.0568**

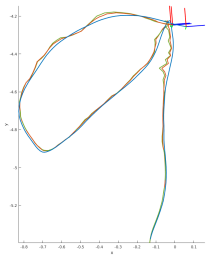
Table 4.2: Mean ATE, Maximum ATE, and RMSE between the ground truth and estimated trajectories. The best results over each sequence are bolded.



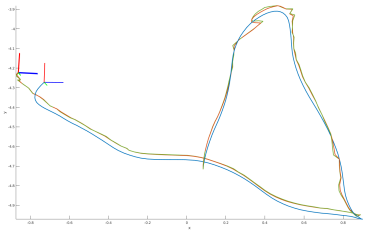
Translational, 3D Plot



Unconstrained 3D Plot



Translational, Projection to XY plane.



Unconstrained, Projection to XY plane.

Figure 4.12: Ground truth (blue), naive estimates (green), and joint segmentation-motion estimated camera motion (orange) for the *swinging-translational* and *swinging-unconstrained* sequences. All plots are in the world coordinate frame. The plotted coordinate axes show the final pose of each trajectory, to illustrate the total drift.

In contrast to the dynamic objects, joint segmentation-motion optimization showed improvement over naive direct alignment in all metrics over both sequences. We observed a 7.6% improvement in the ATE for the translational sequence, and 5.0% improvement in the ATE for the unconstrained sequence. We propose two contributing factors.

The first is the exclusion of pixels with high photometric error through outlier segmentation resulting in an improved estimate of the transform in each step of the joint segmentation-motion estimation.

In particular, this implies that for the case of cameras moving through a dynamic

scene, outlier segmentation in combination with Huber weights may perform better than Huber weights alone.

A second possible factor is that improvement is due largely to continued iterations of direct alignment. As in algorithm 1, the direct alignment step of the joint segmentation-motion optimization takes the initial estimate of the transform, $\hat{\xi}$, from the previous direct alignment step.

Since the estimate of the previous step is within the 'basin of attraction' for the coarse-to-fine initialization, improvement may be the result of continued iterations at the last (highest resolution) pyramid level, and hence equivalent to simply continuing to iterate at this level.

In future work, the effects of each of the proposed factors should be evaluated in isolation, and compared to the total improvement in the camera trajectory estimate, to determined the true source of the improvement.

4.3 Virtual Kitti

The Virtual KITTI (VKITTI) dataset [Gai+16] is a realistic synthetic data-set consisting of rendered video sequences that mimic the video sequences of the well-known KITTI odometry dataset [GLU12], created in the Unity games engine. Scenes are rendered from various camera angles, and under varied lighting and (simulated) weather conditions.

The VKITTI sequences include scenes of dynamic objects, namely vehicles, observed from a moving camera in several settings. Ground truth values for most values of interest to our evaluation are provided.

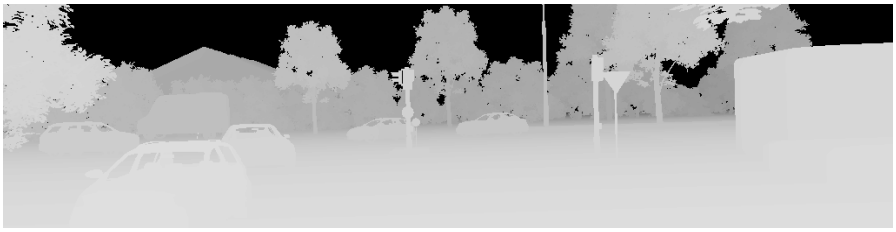
4.3.1 Data

The dataset is made up of several rendered sequences. For each rendered RGB frame, depth and instance segmentation data is provided at a resolution of 1242x375. Examples are shown in figure 4.13.

The rendered frames approximately correspond to the 10 Hz frame-rate of the KITTI dataset of which VKITTI is a simulacrum. Each sequence is rendered with the same fixed intrinsic calibration for the virtual camera, which is provided with the dataset.

Ground truth poses for the camera and for each object in the world are provided in transformation matrix form, in the coordinate frame of the virtual world.

Rendered RGB image.



Registered depth image.

Ground-truth instance segmenation.

Figure 4.13: Example data from the VKITTI dataset. Each sequence includes rendered RGB images, registered depth images, and ground-truth instance segmentation of all objects (including background elements such as street signs).

4.3.2 Preprocessing

The original resolution of the VKITTI sequences is not a power of two, making constructing the 'pyramid' described in section 3.2.4 onerous. We therefore crop the source images to 1024x256.

To do so, we remove 109px from the left and right of each frame. We remove 60px from the top, and 59px from the bottom of each frame. We adjust the intrinsic calibration matrix to account for the new position of the camera center in the image plane.

The dynamic objects in the VKITTI sequences are vehicles, a class of object that is familiar to off-the-shelf instance segmentation neural networks. We use the common Mask-RCNN [He+17] to create instance segmentations of each frame in the sequences of interest.

We use the pre-trained network provided by the authors, and restrict segments to the semantic classes $\{car, truck, van\}$, as they represent the dynamic objects of interest. An example of the ground truth instance segmentation, and the segmentation from Mask-RCNN, is shown in figure 4.14.

The neural network achieves reasonable segmentations. However, we observed a bias to excluding pixels, particularly at the edges of objects (e.g., car mirrors). To include as many pixels belonging to each object as possible, we dilate the instance segmentation produce by the neural network by 3 pixels.

Input RGB image

Ground truth semantic instance segmenation.

Instance mask for classes of interest.

Neural network segmentation.

Instance mask from neural network segmentation.

Figure 4.14: Instance segmentation from an example RGB image. The neural network achieves a reasonable segmentation, however pixels belonging to objects are often missing, particularly at the edges.

Having obtained appropriate instance segmentations and cropped the input images to an appropriate size, we now proceed with our evaluation.

4.3.3 Sequence 18 - Highway

Sequence 18 is a highway sequence with relatively fast moving traffic, as well as oncoming traffic. Though the velocities of the camera and objects relative to the world are large, the relative velocity of the objects to the camera is small, and mainly translational.

We choose a subsequence of 50 frames in which several dynamic objects are visible at the same time. An example RGB image frame is shown in figure 4.15.

Figure 4.15: Sequence 18 Example image.

We estimate the trajectory of the camera and the dynamic objects using the full system, including the joint segmentation-motion optimization.

We construct keyrames at a fixed rate of every 15th frame over the course of the sequence. We retain the activation threshold of $t = 0.3$ as computed in the previous section, as well as the same unary and pairwise segmentation parameters $\sigma = 0.1$, $w = 1.0$.

Since objects may exit the camera's field of view, we implement tracking loss detection. If an active instance label has greater than 70% of the corresponding pixels segmented as outlier, we consider tracking for that instance to have been lost.

The resulting trajectory estimates are plotted in figure 4.16. Additionally, we evaluate the ATE against the ground-truth for each tracked object. These results are tabulated in table 4.3.

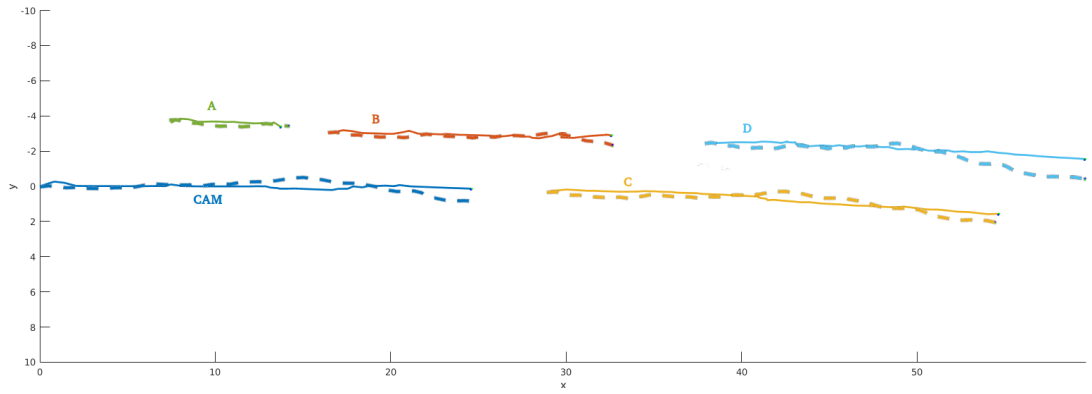


Figure 4.16: Tracks for dynamic objects, as well as the moving camera, for the highway sequence, projected to the X-Y plane. Forward motion of the objects and camera is in the positive X direction. Object tracks are labeled A-D, the camera track is labeled CAM. Ground truth tracks are shown as dashed lines.

Track	**Mean ATE** (m)
Camera	0.544
A	0.218
B	0.417
C	0.423
D	0.934

Table 4.3: Sequence 18 tracks, mean ATE.

Tracking loss occurred for object A (black car nearest the bottom left of the frame) after 23 frames as it moved out of view.

Additionally, tracking loss occurred for object B (white car) after 31 frames. This was due to a large illumination change as the vehicle moved into sunlight, causing large photometric residuals which could not be compensated.

Tracking for object D was poor relative to the other objects. The likely cause is that this object remained furthest from the camera, and thus had relatively few pixels for direct alignment.

The system was unable to track oncoming vehicles, which were visible for only a

few frames. The relative motion for these was large, and therefore the direct alignment portion of the algorithm did not converge to a good estimate.

The system successfully tracked several dynamic objects in this sequence, demonstrating that tracking of dynamic objects from a moving camera with direct photometric alignment is possible, in the case of small relative motions. Each instance was activated after the first frame, and was tracked until tracking loss occurred.

We next evaluate the system's effectiveness on a more complex scene, where instances may activate later than the first frame.

4.3.4 Sequence 20 - Traffic Jam

Sequence 20 is a sequence with slow moving traffic, in a traffic jam. Several objects enter the field of view of the camera later in the sequence, while others leave before the end of the sequence. An example RGB image frame is shown in figure 4.17.

Figure 4.17: Sequence 20 Example image.

We evaluate tracking effectiveness over the entire 200 frame sequence. We estimate the trajectory of the camera and the dynamic objects using the full system, with the same parametrization as for the previous sequence. The velocity of the camera relative to the world is smaller than in sequence 18, and hence we use every 25th frame as a new keyframe.

The resulting trajectory estimates are plotted in figure 4.18. Additionally, we evaluate the ATE against the ground-truth for each tracked object. These results are tabulated in table 4.4.

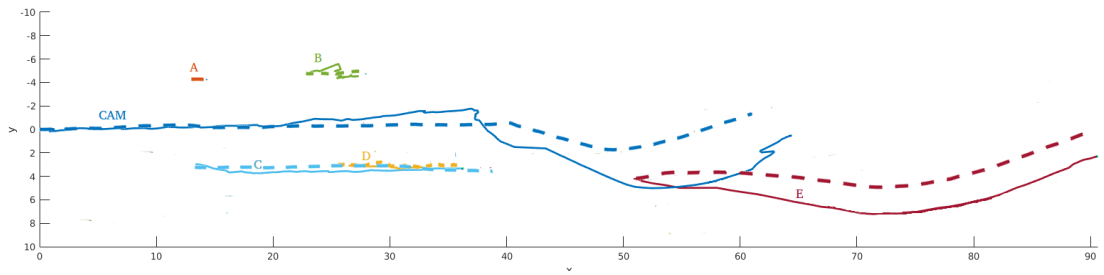


Figure 4.18: Tracks for dynamic objects, as well as the moving camera, for the highway sequence, projected to the X-Y plane. Forward motion of the objects and camera is in the positive X direction. Object tracks are labeled A-D, the camera track is labeled CAM. Ground truth tracks are shown as dashed lines.

Track	**Mean ATE** (m)
Camera	1.173
A	0.533
B	0.709
C	0.497
D	0.505
E	1.711

Table 4.4: Sequence 18 tracks, mean ATE.

Tracking performance was broadly similar to that for sequence 18.

Each object with relative motion was successfully activated in the system, including those which entered the scene at later frames (object E). Tracking loss occurred as objects left the field of view due to camera motion or occlusion by other objects. Tracking loss also occurred for objects as they became more distant, as in sequence 18.

We note that the ATE for the camera and for object E, which entered the field of view later in the sequence, was generally worse than the camera and object tracking in sequence 18. The poorer camera tracking is likely due to accumulated drift over the much longer camera trajectory.

Because object E entered later in the sequence, the accumulated camera drift compounds the existing tracking error.

The system once again successfully tracked several dynamic objects. Additionally, we demonstrated that it is possible to track only moving objects of a given class in the dynamic scene, even though many other stationary objects of that class may be present. Finally, we demonstrate that it is possible to track new objects that were not previously in the camera's field of view.

5 Conclusion

In this thesis we have demonstrated the possibility of using direct photometric alignment to estimate the motion of dynamic objects, from both stationary and dynamic cameras. Additionally, we have demonstrated the use of photometric residuals in identifying dynamic objects within the scene, and the use of instance segmentation for disambiguation. Finally, we have investigated the effectiveness of a joint segmentation-motion estimation strategy, showing that it was of limited advantage over a robust weighting scheme that reduced the influence of outliers, and did not significantly improve initial instance segmentations.

A key finding of this thesis is the fundamental limitation of tracking objects through direct photometric alignment when the relative rotation is large. We have presented several possible explanations, including the possibility of a large induced translational component of the relative motion causing poor convergence. Though we have speculated about the potential causes, the fundamental reason remains unknown. Determining the fundamental cause and mitigating it should be the focus of subsequent research.

We have also identified several other possible directions for future work.

5.1 Future Work

We have identified several potential directions for future research, in the course of the development and evaluation of the algorithm presented in this thesis. These include;

- The system as presented requires a reasonably good initial depth in order to perform motion estimation. This is not generally available, particularly in outdoor scenes. A future direction of work may incorporate depth estimation alongside the motion estimation for dynamic objects. In practice, this would resemble the application of a photometric SLAM to dynamic objects. The full-system optimization approach as in [EKC18] is a useful starting point.

- Though we mentioned it in section 3.2, we did not fully exploit the parallel structure of the problem in this thesis to improve computational efficiency. While implementations of direct photometric alignment have used vectorized instructions to compute the Hessian with an accumulator approach [ESC14] [EKC18], these are limited to the width of admissible vectors and are CPU-bound. This limits the number of points that can be processed in each frame, and hence the available information that can be consumed. This introduces a trade-off between accuracy and computational performance. Furthermore, in this thesis the motion estimate for each active object is computed serially.

  Recently GPUs have become very widely available, including in mobile and low-power applications. A GPU-optimized implenetation could fully exploit the parallelism inherent in each step of the algorithm, thus increasing throughput. This would allow for the use of higher resolution imagery at faster framerates, and a more dense set of points, thus mitigating the accuracy-performance tradeoff. With careful implementation, it would also be possible to estimate the motion for each object in parallel, resulting in significant performance gains.

- Improved tracking performance may be achieved through leveraging the semantic information of each instance. In particular, the use of shape priors and motion models for pose and trajectory optimization of dynamic objects as in [ESL17] and [ESL16] as priors for direct photometric estimation is likely to yield accuracy improvements for certain classes of interest, such as street vehicles.

- Imposing temporal consistency on the segmentation, for example through extending the CRF formulation to associate points over multiple frames, similar to [SB15], may improve segmentation performance over time. The ambiguity problem discussed in section 3.4 may be resolved through accumulating an object segmentation over multiple frames, without the need for an initial instance segmentation.

- The system as presented is a fully dense approach, using all available pixels in the image. However, a sparse formulation similar to [EKC18] leveraging only points with sufficient texture may lead to reduced computational requirements, as well as reduced noise from regions of the image frame that carry little information.

- We use a first-order photometric residual throughout. However, the use of a second-order residual which computes edge-wise photometric consistency (i.e.

that pixels at image edges should remain at image edges in subsequent frames) may also improve tracking, and aid in disambiguation.

A fundamental limitation to further research in the direction of dynamic photometric odometry is the lack of suitable datasets. Both datasets used for evaluation in this thesis had key limitations, missing ground truth data as in OMMD, or else limited realism and variety of motions as in VKITTI. The further elaboration of a real-world dataset, or an improved synthetic dataset, would allow for improved evaluations and a deeper understanding of the algorithm's limitations.

In conclusion, this thesis demonstrates the possibility of estimating the motions of dynamic objects, as well as the motion of the camera, through photometric odometry. We have identified identified key limitations that indicate the most important directions for future work. Finally, we have demonstrated a proof of concept that successfully identifies and tracks dyanamic objects in representative environments.

List of Figures

List of Tables

Bibliography

[AA05] M. K. Agoston and M. K. Agoston. *Computer graphics and geometric modeling*. Vol. 1. Springer, 2005.

[Ant14] A. Anton. *MATLAB wrapper for Boykov-Kolmogorov max-flow/min-cut algorithm.* Oct. 2014.

[ARS10] M. Andriluka, S. Roth, and B. Schiele. "Monocular 3d pose estimation and tracking by detection." In: *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE. 2010, pp. 623–630.

[Bar+17] D. Barnes, W. Maddern, G. Pascoe, and I. Posner. "Driven to Distraction: Self-Supervised Distractor Learning for Robust Monocular Visual Odometry in Urban Environments." In: *arXiv preprint arXiv:1711.06623* (2017).

[Bâr+18] I. A. Bârsan, P. Liu, M. Pollefeys, and A. Geiger. "Robust Dense Mapping for Large-Scale Dynamic Environments." In: (2018).

[Bes+18] B. Bescós, J. M. Fácil, J. Civera, and J. Neira. "DynSLAM: Tracking, Mapping and Inpainting in Dynamic Scenes." In: *CoRR* abs/1806.05620 (2018).

[Bis06] C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Berlin, Heidelberg: Springer-Verlag, 2006. ISBN: 0387310738.

[BK04] Y. Boykov and V. Kolmogorov. "An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision." In: *IEEE transactions on pattern analysis and machine intelligence* 26.9 (2004), pp. 1124–1137.

[Bre+09] M. D. Breitenstein, F. Reichlin, B. Leibe, E. Koller-Meier, and L. Van Gool. "Robust tracking-by-detection using a detector confidence particle filter." In: *2009 IEEE 12th International Conference on Computer Vision*. IEEE. 2009, pp. 1515–1522.

[Che+18] J. Cheng, Y. Sun, W. Chi, C. Wang, H. Cheng, and M. Q.-H. Meng. "An Accurate Localization Scheme for Mobile Robots Using Optical Flow in Dynamic Environments." In: *2018 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE. 2018, pp. 723–728.

[Cor+16] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. "The Cityscapes Dataset for Semantic Urban Scene Understanding." In: *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.

[EKC18] J. Engel, V. Koltun, and D. Cremers. "Direct sparse odometry." In: *IEEE transactions on pattern analysis and machine intelligence* 40.3 (2018), pp. 611–625.

[ESC14] J. Engel, T. Schöps, and D. Cremers. "LSD-SLAM: Large-scale direct monocular SLAM." In: *European Conference on Computer Vision*. Springer. 2014, pp. 834–849.

[ESL16] F. Engelmann, J. Stückler, and B. Leibe. "Joint Object Pose Estimation and Shape Reconstruction in Urban Street Scenes Using 3D Shape Priors." In: *Proc. of the German Conference on Pattern Recognition (GCPR)*. 2016.

[ESL17] F. Engelmann, J. Stückler, and B. Leibe. "SAMP: Shape and Motion Priors for 4D Vehicle Reconstruction." In: *IEEE Winter Conference on Applications of Computer Vision, WACV*. 2017.

[Gai+16] A. Gaidon, Q. Wang, Y. Cabon, and E. Vig. "Virtual Worlds as Proxy for Multi-Object Tracking Analysis." In: *CVPR*. 2016.

[GLU12] A. Geiger, P. Lenz, and R. Urtasun. "Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite." In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2012.

[Goo+14] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. "Generative adversarial nets." In: *Advances in neural information processing systems*. 2014, pp. 2672–2680.

[GPS89] D. M. Greig, B. T. Porteous, and A. H. Seheult. "Exact Maximum A Posteriori Estimation for Binary Images." In: *Journal of the Royal Statistical Society. Series B (Methodological)* 51.2 (1989), pp. 271–279. ISSN: 00359246.

[He+17] K. He, G. Gkioxari, P. Dollár, and R. Girshick. "Mask r-cnn." In: *Computer Vision (ICCV), 2017 IEEE International Conference on*. IEEE. 2017, pp. 2980–2988.

[Hub11] P. J. Huber. *Robust statistics*. Springer, 2011.

[JG19] K. M. Judd and J. Gammell. "The Oxford Multimotion Dataset: Multiple SE (3) Motions with Ground Truth." In: *IEEE Robotics and Automation Letters* (2019).

[JGN18] K. M. Judd, J. D. Gammell, and P. Newman. "Multimotion Visual Odometry (MVO): Simultaneous Estimation of Camera and Third-Party Motions." In: *CoRR* abs/1808.00274 (2018).

[KFB09] D. Koller, N. Friedman, and F. Bach. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

[KGB16] A. Kurakin, I. Goodfellow, and S. Bengio. "Adversarial examples in the physical world." In: *arXiv preprint arXiv:1607.02533* (2016).

[KM09] G. Klein and D. Murray. "Parallel tracking and mapping on a camera phone." In: *2009 8th IEEE International Symposium on Mixed and Augmented Reality*. IEEE. 2009, pp. 83–86.

[KSC13] C. Kerl, J. Sturm, and D. Cremers. "Dense visual SLAM for RGB-D cameras." In: *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*. Citeseer. 2013, pp. 2100–2106.

[Li+04] L. Li, W. Huang, I. Y.-H. Gu, and Q. Tian. "Statistical modeling of complex backgrounds for foreground object detection." In: *IEEE Transactions on Image Processing* 13.11 (2004), pp. 1459–1472.

[Liu+16] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. "Ssd: Single shot multibox detector." In: *European conference on computer vision*. Springer. 2016, pp. 21–37.

[Mar63] D. W. Marquardt. "An algorithm for least-squares estimation of nonlinear parameters." In: *Journal of the society for Industrial and Applied Mathematics* 11.2 (1963), pp. 431–441.

[MT17] R. Mur-Artal and J. D. Tardós. "Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras." In: *IEEE Transactions on Robotics* 33.5 (2017), pp. 1255–1262.

[NLD11] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison. "DTAM: Dense tracking and mapping in real-time." In: *2011 international conference on computer vision*. IEEE. 2011, pp. 2320–2327.

[NP14] G. Nebehay and R. Pflugfelder. "Consensus-based matching and tracking of keypoints for object tracking." In: *IEEE Winter Conference on Applications of Computer Vision*. IEEE. 2014, pp. 862–869.

[Ose+17] A. Osep, W. Mehner, P. Voigtlaender, and B. Leibe. "Track, then Decide: Category-Agnostic Vision-based Multi-Object Tracking." In: *CoRR* abs/1712.07920 (2017). arXiv: 1712.07920.

[Ozu+10] M. Ozuysal, M. Calonder, V. Lepetit, and P. Fua. "Fast keypoint recognition using random ferns." In: *IEEE transactions on pattern analysis and machine intelligence* 32.3 (2010), pp. 448–461.

[PD11] H. Poon and P. M. Domingos. "Sum-product networks: A new deep architecture." In: *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)* (2011), pp. 689–690.

[Red+16] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. "You only look once: Unified, real-time object detection." In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 779–788.

[SB15] J. Stückler and S. Behnke. "Efficient dense rigid-body motion segmentation and estimation in RGB-D video." In: *International Journal of Computer Vision* 113.3 (2015), pp. 233–245.

[Sco+18] R. Scona, M. Jaimez, Y. R. Petillot, M. Fallon, and D. Cremers. "StaticFusion: Background Reconstruction for Dense RGB-D SLAM in Dynamic Environments." In: *Robotics and Automation (ICRA), 2018 IEEE International Conference on*. IEEE. 2018.

[SCR12] R. Samdani, M.-W. Chang, and D. Roth. "Unified expectation maximization." In: *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics. 2012, pp. 688–698.

[SS05] Y. Sheikh and M. Shah. "Bayesian modeling of dynamic scenes for object detection." In: *IEEE transactions on pattern analysis and machine intelligence* 27.11 (2005), pp. 1778–1792.

[Stu+12] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. "A Benchmark for the Evaluation of RGB-D SLAM Systems." In: *Proc. of the International Conference on Intelligent Robot Systems (IROS)*. Oct. 2012.

[TI14] D. J. Tan and S. Ilic. "Multi-forest tracker: A chameleon in tracking." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2014, pp. 1202–1209.

[V+01] P. Viola, M. Jones, et al. "Robust real-time object detection." In: *International journal of computer vision* 4.34-47 (2001), p. 4.

[Wer+09] M. Werlberger, W. Trobin, T. Pock, A. Wedel, D. Cremers, and H. Bischof. "Anisotropic Huber-L1 Optical Flow." In: *BMVC*. Vol. 1. 2. 2009, p. 3.

[Xu+18] B. Xu, W. Li, D. Tzoumanikas, M. Bloesch, A. J. Davison, and S. Leutenegger. "MID-Fusion: Octree-based Object-Level Multi-Instance Dynamic SLAM." In: *CoRR* abs/1812.07976 (2018). arXiv: 1812.07976.

[YMU14] K. Yamaguchi, D. McAllester, and R. Urtasun. "Efficient Joint Segmentation, Occlusion Labeling, Stereo and Flow Estimation." In: *ECCV*. 2014.

[ZBS01] Y. Zhang, M. Brady, and S. Smith. "Segmentation of brain MR images through a hidden Markov random field model and the expectation-maximization algorithm." In: *IEEE transactions on medical imaging* 20.1 (2001), pp. 45–57.

[ZJD00] Y. Zhong, A. K. Jain, and M.-P. Dubuisson-Jolly. "Object tracking using deformable templates." In: *IEEE transactions on pattern analysis and machine intelligence* 22.5 (2000), pp. 544–549.

[ZLY12] W. Zhong, H. Lu, and M.-H. Yang. "Robust object tracking via sparsity-based collaborative model." In: *2012 IEEE Conference on Computer vision and pattern recognition*. IEEE. 2012, pp. 1838–1845.