Exploiting Rigid Body Motion for SLAM in Dynamic Environments

Mina Henein, Gerard Kennedy, Robert Mahony and Viorela Ila

Abstract— **The limitations of existing localisation and mapping algorithms in handling highly dynamic environments is a key roadblock in the deployment of autonomous mobile robotic systems in a range of important real world situations. In this paper we propose a technique to integrate the motion of dynamic objects into a Simultaneous Localisation and Mapping (SLAM) algorithm without the need to know *a-priori* or model the geometry of the object, or even to explicitly estimate the pose of the object. The benefit of this approach lies in a simplification of the underlying SLAM state and a resulting simplification of the non-linear least squares optimisation solution. We demonstrate the performance of the algorithm on two scenarios; SLAM in an urban traffic scenario, and extrinsic calibration of a multi RGBD camera system observing a moving object. Our experiments show consistent improvement in robot localisation and mapping accuracy and demonstrate potential of the proposed algorithm.**

I. Introduction

The world is a complex and highly dynamic environment, and thus to allow robotics to be part of our daily lives, research in autonomous robotics is rapidly departing from simple, controlled environments to environments that are more representative of the reality. In order to perform tasks autonomously in such environments, a robot must be able to simultaneously build an accurate representation of its surroundings and localise itself. Simultaneous localisation and mapping (SLAM) algorithms are a core enabling technology for autonomous mobile robotics.

SLAM is a well researched area in robotics, and while many efficient solutions to the problem exist, most of the existing techniques heavily rely on the static world assumption [1]; an assumption that is invalid in highly dynamic environments. The conventional technique in SLAM for dealing with dynamics is to either treat them as outliers [2], [3], [4], [5] or detect and track them separately using traditional multi-target tracking [6], [7], [8]. In neither case is the information about the moving points used in the robot pose estimate within the SLAM algorithm. This means that in dynamic environments, the resulting SLAM problem can easily fail due to the lack of reliable static environment features [9], [10]. Although point representation is the most commonly used method of representing the environment in visual SLAM [11], primitives such as lines, planes [12], [13], [14], [15] and even objects [16], [17] can provide richer map representations, and are better suited to SLAM in dynamic environments [18], [19].

The authors are with the Australian Centre of Excellence for Robotic Vision, Research School of Engineering, The Australian National University, Canberra ACT 2601, Australia mina.henein, gerard.kennedy, robert.mahony and viorela.ila@anu.edu.au

Existing object SLAM algorithms require a two stage approach; initially the objects are segmented, tracked and their poses are estimated (the 'front end'), and then this information is used to estimate the state in a probabilistic framework (the 'back end') [11]. The problem of object detection and segmentation seems to have achieved accurate solutions at almost real time [20], [21]. However, to the best of our knowledge, the problem of 3D pose estimation still remains a challenge to learning techniques [22], [23] that are incorporated into state of the art SLAM front-ends. Existing object or primitive structure-based SLAM algorithms suffer from the inability of the front end to provide reliable pose/structure information, even though the segmentation, that associates pixels to objects, is highly reliable [20], [21].

In this paper, we propose a feature-based algorithm that exploits data segmentation to integrate rigid body motion of objects into a SLAM framework, without the need to estimate the object pose or to have prior knowledge of the object's 3D model. We use *constant motion* to model the motion of dynamic objects in the environment. We incorporate this information into the SLAM framework by a simple change of variables that relies on the rigidity assumption of the moving objects in the scene. We evaluate our algorithm on two applications; driving in an urban environment, and extrinsic calibration (estimation of relative camera poses) of a multi RGBD camera system. In highly dynamics environments our algorithm produces consistently better results compared to accepted algorithms that exclude moving features from the robot pose SLAM estimation. In addition, the algorithm is directly applicable to the problem of multi-camera extrinsic calibration, even in the case where the cameras have non-overlapping views, as long as there is an observed object that moves with constant motion (for example circling the camera array) that is seen by all cameras.

The remainder of this paper is structured as follows, in the following section we discuss the related work. In section III we describe the proposed approach for incorporating dynamics of the scene. In section IV we introduce the experimental setup, followed by the experimental results and evaluations in section V. We summarise and offer concluding remarks in section VI.

II. Related Work

The earliest work on SLAM was based on the extended Kalman filter (EKF) approach [24], [25]. However, it has been shown that filtering is inconsistent when applied to the inherently non-linear SLAM problem [26]. One intuitive way of formulating SLAM is to use a graph representation [27]

with nodes corresponding to the random variables (robot poses and/or landmarks in the environment) and edges representing functions of those variables (typically measurement functions). Once such a graph is constructed, the goal is to find a configuration of the nodes that is maximally consistent with the measurements [28]. Approaching SLAM as a non-linear optimisation on graphs has been shown to offer very efficient solutions to SLAM applications [29], [30], [31].

In SLAM algorithms, information associated with stationary objects is considered positive, while moving objects are seen to degrade the performance, and are either treated as outliers [2], [3], [4], [5] or tracked separately using multi-target tracking [6], [7], [8]. Conversely, measurements belonging to moving objects are required for moving object tracking algorithms, while stationary points and objects are considered background and filtered out. However, establishing the spatial and temporal relationships between a robot, stationary objects, and moving objects in a scene serves as a basis for scene understanding [18]. Simultaneous localisation, mapping and moving object tracking are therefore mutually beneficial.

One of the earliest work in the area of SLAM in dynamic environments is presented by Hahnel *et al.* [3] who use an Expectation-Maximisation (EM) algorithm to update the probabilistic estimate about which measurements corresponded to a static/dynamic object and remove them from the estimation when they correspond to a dynamic object. Bibby and Reid's SLAMIDE [32] also estimates the state of 3D features (stationary or dynamic) with a generalised EM algorithm where they use reversible data association to include dynamic objects into a single framework SLAM. Wang *et al.* [18] developed a theory for performing SLAM with Moving Objects Tracking (SLAMMOT). They first presented a SLAM algorithm with generalised objects, which computes the joint posterior over all objects and the robot, an approach that is computationally demanding and generally infeasible as stated by the authors. They also developed a SLAM algorithm with detection and tracking of moving objects, in which the estimation problem is decomposed into two separate estimators, for the stationary and moving objects, which results in a much lower dimensionality and makes it feasible to update both filters in real time. Our algorithm, described in the next section, exploits rigid body motion in a single graph-SLAM framework.

III. Accounting for Dynamic Objects in SLAM

A. Notation

We use factor graphs [33] to model the SLAM with dynamic objects estimation problem. Camera/robot poses are parameterised by $\mathbf{x} = \{x_0 ... x_{n_x}\}$, with $x_k \in \mathfrak{se}(3)$, and n_x is the number of time steps. That is, the pose of the robot at time step k is $\exp(x_k) \in SE(3)$. The sequence of 3D point features are denoted by $\mathbf{l} = \{l_0^1 \dots l_{n_x}^{n_l}\}$ with $l_k^i \in \mathbb{R}^3$ and $i \in 1...n_l$ is the unique index of a landmark and n_l is the total number of detected landmarks. This constitutes the set of variables in a typical SLAM problem. The set of landmarks, $\mathbf{l} = \mathbf{l_s} \cup \mathbf{l_d}$, constitutes of a set of static landmarks $\mathbf{l_s}$ and a set of landmarks detected on moving objects at different time steps, $\mathbf{l_d}$. Let $\{0\}$ denote the *reference coordinate frame*. The robot/camera poses and the positions of the 3D points are represented in the $\{0\}$ reference frame. For a SLAM problem with dynamic rigid body objects, let $\{L\}$ be a coordinate frame associated to a moving rigid body (object). ${}^0L_k \in SE(3)$ is the rigid body pose with respect to the reference frame $\{0\}$, ${}^0b_k \in \mathfrak{se}(3) | {}^0b_k = \log({}^0L_k)$. For a feature observed on an object, ${}^Ll^i \in \mathbb{R}^3$ denotes the coordinates of this point in the body-fixed frame, where 'i' is the unique index of the feature. We write ${}^0l_k^i$ the coordinates of the same point expressed in the reference frame $\{0\}$ at time k. The relative motion of the object from time k to time $k+1$ is represented by a rigid-body transformation ${}^0_kH^j_{k+1} \in SE(3)$, with ${}^0_ku^j_{k+1} \in \mathfrak{se}(3) | {}^0_ku^j_{k+1} = \log({}^0_kH^j_{k+1})$, $j \in 1...n_o$ is the object index and n_o is the number of identified objects.

B. Graph formulation

We start from a 'naive' way of integrating rigid body motion into the estimation, with the aim of justifying why this implementation is impractical and in most cases infeasible. Fig. 1 shows a factor graph representation of an implementation of a SLAM algorithm that integrates dynamic body motion into the estimation. In Fig. 1, solid black nodes represent random variables to be estimated, and coloured dots represent the factors relating those nodes. Blue factors represent odometric measurements, red factors represent point measurements in camera frames, green factors represent relative positions of points with respect to the object pose they belong to, and grey factors represent the object motion constraints. We are particularly interested in the relative point positions with respect to their object pose and the object motion factors.

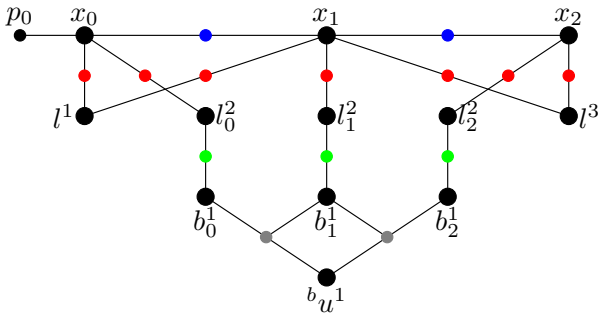


Fig. 1: Factor graph representation of a SLAM problem with moving objects.

The green factors encode the positions of the landmarks in the body fixed frame, which are constant for every time step (assuming rigidity of the objects). Implementation of these factors requires some knowledge of the object 3D model, either estimated at every time step or known *a-priori*. The grey factors require an measurement of the object pose at every time step. This is usually provided by the front end explicitly based on a model of the point estimates or directly from a deep network or similar front end.

A-priori knowledge of 3D models of the objects in the scene requires development of an object database beforehand [17] or training of a deep network on known example scenarios. Even if such prior learning is available, state-of-the-art front-end algorithms are still unable to reliably and accurately provide object poses/motion [22], [23]. In many real-world situations the ability to undertake prior learning is not available and algorithms that rely on such information are infeasible.

C. Motion model of a point on a rigid body

This section shows how the motion of the rigid body objects can directly be transferred to the motion of the feature points without the need of estimating for the pose or the geometry of the moving object.

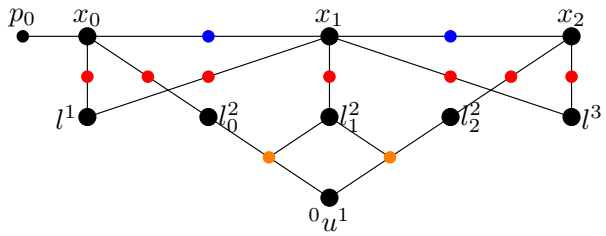


Fig. 2: Factor graph representation of a problem with a motion vertex and its edges.

Fig. 2 shows a factor graph representation of a SLAM problem that integrates the motion of a rigid body object that acts directly on the feature points without the need to estimate the rigid body pose. First, we write the relative position of a landmark with respect to the object centroid for two consecutive time steps; k and $k+1$. Using the assumption that the object in question is a rigid body, it follows that ${}^L\bar{l}^i_k = {}^L\bar{l}^i_{k+1}$ which can be written as;

$$ {}^0L_k^{-1}\,{}^0\bar{l}^i_k = {}^0L_{k+1}^{-1}\,{}^0\bar{l}^i_{k+1} \tag{1} $$

where ${}^L\bar{l}^i$ and ${}^0\bar{l}^i_k$ are the homogeneous coordinates of the points ${}^Ll^i$ and ${}^0l^i_k$, respectively. Factor 'c' represents the relation between the position of a landmark with respect to the parameterisation of its object centroid is shown in Fig. 3a. From which,

$$ {}^0\bar{l}^i_{k+1} = {}^0L_{k+1}\,{}^0L_k^{-1}\,{}^0\bar{l}^i_k \tag{2} $$

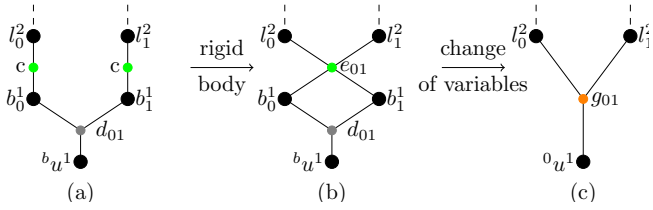


Fig. 3: Transition from a factor graph representation with rigid body object poses to one with no object poses and with a motion vertex

The factor corresponding to (2) represents a quaternary factor that relates the same landmark at two consecutive time steps, and the poses of the object on which it belongs at these two time steps, and is defined as follows;

$$ \begin{aligned} &e({}^0l^i_k, {}^0l^i_{k+1}, {}^0b_k, {}^0b_{k+1}) := \\ &\quad {}^0R_{b_k} * {}^0R_{b_{k+1}}^{\top} * ({}^0l^i_{k+1} - {}^0t_{b_{k+1}}) + {}^0t_{b_k} - {}^0l^i_k + a^j_k \end{aligned} \tag{3} $$

where with ${}^0R_{b_k}$ and ${}^0t_{b_k}$ are obtained using the exponential map $\exp({}^0b_k)$, and $a_k \sim \mathcal{N}(0, \Sigma_{a_k})$ normally distributed with zero mean Gaussian noise with covariance Σ_{a_k}. Fig. 3 shows the transition from a factor graph representation with rigid body objects (Fig. 3a) to one with motion vertices and no object poses (Fig. 3b), passing by one with a quaternary factor between two landmarks at consecutive time steps and their respective object poses at the same time steps (Fig. 3c). Factor 'd' represents the relation between the parameterisation of two consecutive object poses and their respective body-fixed frame pose change is shown in Fig. 3b. Using the fact that ${}^0L_{k+1}$ can be written as ${}^0L_{k+1} = {}^0L_k\,{}^k_kH_{k+1}$, where ${}^k_kH_{k+1} \in SE(3)$ (which we call *body-fixed frame pose change*) is the rigid body transformation that represents the relative motion of the object from time k to time $k+1$ expressed in body-fixed frame, and with a simple change of variables, (2) can be re-written as

$$ {}^0\bar{l}^i_{k+1} = {}^0L_k\,{}^k_kH_{k+1}\,{}^0L_k^{-1}\,{}^0\bar{l}^i_k = {}^0_kH_{k+1}\,{}^0\bar{l}^i_k \tag{4} $$

where ${}^0_kH_{k+1} = {}^0L_k\,{}^k_kH_{k+1}\,{}^0L_k^{-1} \in SE(3)$. According to [34], this equation represents a frame change of a pose transformation, and shows how the body-fixed frame pose change relates to the *reference frame pose change*. The factor corresponding to (4) and shown in Fig. 3c is:

$$ g(l^i_k, l^i_{k+1}, {}^0_ku^j_{k+1}) = {}^0_kR^j_{k+1}{}^{\top}\,{}^0l^i_{k+1} - {}^0_kR^j_{k+1}{}^{\top}\,{}^0_kt^j_{k+1} - {}^0l^i_k + q_{s_j} \tag{5} $$

where ${}^0_kR^j_{k+1}$ and ${}^0_kt^j_{k+1}$ are obtained using the exponential map $\exp({}^0_ku^j_{k+1})$, and $q_s \sim \mathcal{N}(0, \Sigma_q)$ is the normally distributed zero-mean Gaussian noise. The factor in (5) is a ternary factor that we call the *motion model of a point on a rigid body*.

This formulation is key to the proposed approach since *it eliminates the need to estimate the actual object pose* and allows us to work directly with points ${}^0l^i_k$ in the reference frame. A second key observation we make in this paper is that if the body-fixed frame pose change is constant between time steps then the reference frame pose change is also constant. This can be seen by replacing ${}^0L_k$ by ${}^0L_{k-1}\,{}^{k-1}_{k-1}H_k$ in ${}^0_kH_{k+1} = {}^0L_k\,{}^k_kH_{k+1}\,{}^0L_k^{-1}$ to obtain:

$$ {}^0_kH_{k+1} = {}^0L_{k-1}\,{}^{k-1}_{k-1}H_k\,{}^0L_{k-1}^{-1} = {}^0_{k-1}H_k \tag{6} $$

Therefore, for a rigid-body object in motion we can use a constant reference frame pose change $H \in SE(3)$ that acts on the points on the rigid body to update their reference frame coordinates: ${}^0\bar{l}^i_{k+1} = H\,{}^0\bar{l}^i_k$. This then allows us to draw the factor graph representation as shown in Fig. 2.

The set of all variables is now $\theta = \mathbf{x} \cup \mathbf{l} \cup \mathbf{u}$, where $\mathbf{u}$ is the set of all the variables characterising the objects' motion.

D. Measurements and object motion constraints

Two types of measurements, the odometry obtained from the robot's proprioceptive sensors, and the observations of the landmarks in the environment obtained by processing the images from an on-board camera are typically integrated into a visual SLAM application. Let $f(x_{k-1}, x_k)$ be the odometry model with Σ_{v_k}, odometry noise covariance matrix:

$$o_k = f(x_{k-1}, x_k) + v_k\,, \quad \text{with} \quad v_k \sim \mathcal{N}(0, \Sigma_{v_k})\,, \tag{7}$$

and $\mathbf{o} = \{o_1 ... o_{m_i}\}$ being the set of m_i odometric measurements. Similarly, let $h(x_k, l_k^i)$ be the 3D point measurement model with Σ_{w_k}, the measurement noise covariance matrix:

$$z_k^i = h(x_k, l_k^i) + w_k^i\,, \quad \text{with} \quad w_k^i \sim \mathcal{N}(0, \Sigma_{w_k}) \tag{8}$$

where $\mathbf{z} = \{z_1 ... z_{m_k}\}$, $z_k \in \mathbb{R}^3$ is the set of all 3D point measurements at all time steps.

The relative pose transformation of the points on moving rigid bodies is given in (4). Observe that the motion of any point on a specific object j can be characterised by the same pose transformation ${}^0_k H^j_{k+1} \in SE(3)$ with ${}^0_k R^j_{k+1}$ the rotation component and ${}^0_k t^j_{k+1}$ the translation component, respectively.

In this paper we show results of integrating constant motion into the SLAM estimation problem. Fig. 2 depicts the factor graph of a small SLAM example of three robot poses, two static features and a feature detected at three different time steps on an object with constant motion. We say that a pose change ${}^0_k H^j_{k+1}$ is constant for all the points on an object j at every time step, hence a sole state variable ${}^0u^j$ is used for each object and the factor in (5) becomes:

$$g(l_k^i, l_{k+1}^i, {}^0u^j) = {}^0R^{j\top}\,{}^0l_{k+1}^i - {}^0R^{j\top}\,{}^0t^j - {}^0l_k^i + q_{s_j} \tag{9}$$

E. The graph optimisation

Given the measurements and motion factors introduced above, we can formulate an NLS problem to obtain the optimal solution of the SLAM with dynamic objects:

$$\theta^* = \underset{\theta}{\operatorname{argmin}}\, \frac{1}{2}\left\{ \sum_{i=1}^{mi} \|f(x_{k-1}, x_k) - o_k\|^2_{\Sigma_{v_k}} + \right.$$
$$\left. \sum_{k=1}^{mk} \|h(x_i, l^i) - z_k\|^2_{\Sigma_{w_k}} + \sum_{i,j}^{ms} \|g(l_k^i, l_{k+1}^i, {}^0u^j)\|^2_{\Sigma_q} \right\} \tag{10}$$

where *mi*, *mk* and *ms* are the number of odometric measurements, point measurement and constant motion factors.

Iterative methods such as Gauss-Newton (GN) or Levenberg-Marquard (LM) are used to find the solution of the NLS in (10). An iterative solver starts with an initial point θ^0 and, at each step, computes a correction δ towards the solution. For small $\|\delta\|$, a Taylor series expansion leads to linear approximations in the neighbourhood of θ^0 and a linear system $A^\top A\delta = -A^\top \mathbf{b}$ is solved at each iteration [35]. In here, A gathers the derivatives of all the factors in (10) with respect to variables in θ weighted by the square rooted covariances of each factor, and $\mathbf{b}$ is the residual evaluated at the current linearisation point. The new linearisation point θ^{i+1} is obtained by applying the increment δ^* to the current linearisation point θ^i. This formulation is often used in the SLAM literature [30], [33], [35], [36].

The factor graph formulation of the SLAM problem is highly intuitive and has the advantage that it allows for efficient implementations of batch [33] [37] and incremental [38], [39], [40] NLS solvers.

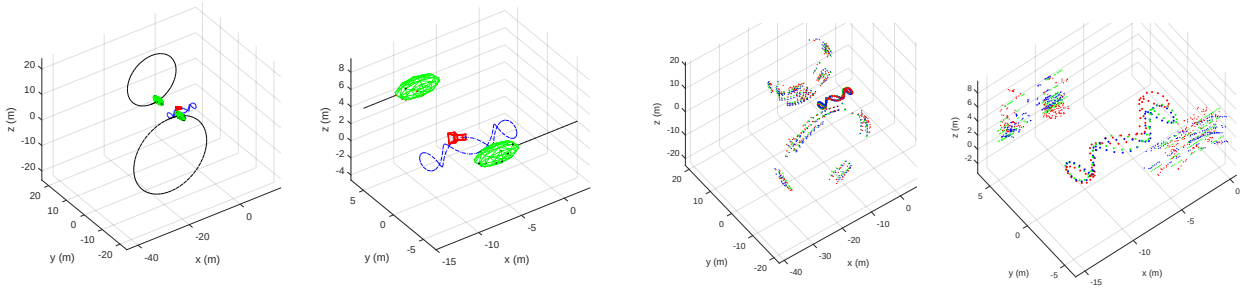
IV. Experimental methodology

Two main problems are considered, and to which a solution is found by employing the same approach explained in III. The first problem consists of a SLAM with moving object tracking, where the goal is to estimate for the robot and moving object trajectories, and the structure of static features in the environment. A second application with a very similar underlying factor graph representation is a multi-camera extrinsic calibration problem that integrates moving objects. A factor graph representation for this problem is similar to the one shown in Fig. 2, however without the presence of odometric measurements (blue factors).

A set of experiments are carried out on simulated data obtained by emulating an advanced front-end that is capable of identifying objects, and associating detected landmarks with the different objects in the scene at every time step. In these simulations, two other types of measurements are also available; odometric measurements and point observations. A second, more realistic experiment, is performed on the Virtual KITTI dataset [41]. Furthermore, three more experiments are carried out for extrinsic multi-camera calibration with different number of cameras. The first two experiments are simulated datasets with 3 non-overlapping field of view cameras, and 4 cameras with minimal overlap. Both experiments provide ground-truth. The third is performed on a real dataset, captured by moving an object around three non-overlapping ZED-cameras mounted on a rig.

A. Experiments

1) Experiments 'A' & 'B': In order to evaluate the effect of integrating the motion of rigid body objects into the SLAM problem, 2 experiments with multiple moving objects are generated using a simulated environment. In both experiments, (we will refer to these as "Experiments 'A' and 'B' in the remainder of this document), the objects are constrained to follow a constant motion trajectory. In Experiment 'A' the objects are following an elliptical trajectory as seen in Fig. 4a. Experiment 'A' is composed of 581 vertices, 910 edges when motion is integrated into the estimation, and 576 edges when no motion is integrated. In Experiment 'B', the objects are only translating in 3D as seen in Fig. 4b. Experiment 'B' is composed of 302 vertices, 460 edges when motion is integrated into the estimation, and 299 edges when no motion is integrated. The decrease in number of edges when no motion is integrated into the estimation is due of the removal of ternary edges. It is worth mentioning that these two experiments involve moving only scene (no static features) and a moving observer.



(a) Experiment 'A' (b) Experiment 'B' (c) Experiment 'A' Solution (d) Experiment 'B' Solution

Fig. 4: Different simulated robot and object trajectories(left). Solution of the SLAM with dynamic objects(right). Large dots represent robot trajectory and small dots represent moving points trajectories. Green - ground truth, Red - the SLAM solution without incorporating object motion information and Blue - the SLAM solution with constant motion. (best viewed in colour)

2) Simulated camera calibration datasets: Two simulated experiments with a multi RGBD camera system were run. In the first experiment, a simulated multi-camera rig consisting of 3 RGBD cameras with completely disjoint field of views with an ellipsoid circumnavigating the rig of 3 cameras, as shown in Fig. 5a, is used to apply our algorithm for the extrinsic calibration of a multi-camera system. The visibility map for each camera is shown in Fig. 5b and shows completely disjoint fields of view. The second experiment features a system of 4 RGBD cameras where cameras 1 & 2 share some overlapping fields of view, as do cameras 3 & 4, as shown in Fig. 5c. The visibility map for each camera is shown in Fig. 5d

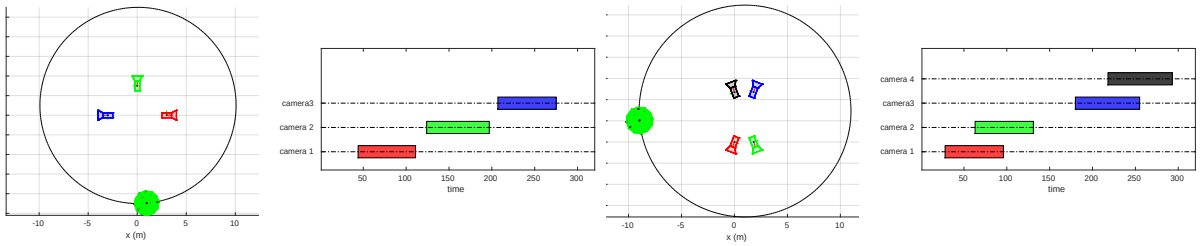
3) Real camera calibration dataset: The real data is acquired by manually moving an object (a checker board pattern) around a rig on which 3 cameras are mounted looking outwards with non-overlapping fields of view. We also make use of the planarity information of the object to enhance the estimation, as explained in [14]. Fig. 6 depicts a snapshot of the real data setup. This experiment will be referred to as the 'real camera calibration' experiment in the remainder of this paper. The 'real camera calibration' experiment is composed of 1104 vertices, and 3078 edges. It is worth mentioning that all camera calibration experiments involve moving only scene (no static features) and a static observer.

4) Virtual KITTI dataset: Virtual KITTI [41] is a photo-realistic synthetic dataset designed to evaluate computer vision scene understanding algorithms. The videos are fully annotated at the pixel level with object labels, and the depth map of each image is also available. This makes it a perfect dataset to test and evaluate the proposed technique on realistic scenarios. Our front-end detects features in each image and obtains the 3D position relative to the camera of each point. The pixel-level object tracking is used to determine if these points are located on moving or static objects, and the known camera poses are used to project the points to the next image in the sequence. In this manner, static and moving points are tracked between images to provide data associations between landmarks and objects, as shown in Fig. 7. It is possible to project points attached to moving objects to other images as the pose of all moving objects is provided by the dataset for each image. As the 3D position is tracked for all the points, along with the camera poses, the relative position of all points can be obtained for every image that the point remains in the camera's field of view. An experiment was run over a total of 1539 vertices, including robot poses, static and dynamic landmark positions and constant motion vertices resulting in a total 3810 edges, of which 1051 are object motion related ternary factors. The noise levels added to the ground truth data in order to generate noisy measurements are as follows: $\Sigma_v = \text{diag}[0.4m, 0.4m, 0.4m, 6^\circ, 6^\circ, 6^\circ]^2$, $\Sigma_w = \text{diag}[0.4m, 0.4m, 0.4m]^2$, $\Sigma_q = \text{diag}[0.05m, 0.05m, 0.05m]^2$. This experiment will be referred as 'vKITTI' in the remainder of this paper, and involves static and dynamic scene along with a moving observer.

B. Implementation details

This work is an extension of an already existing MATLAB framework that is able to integrate not only simple point measurements but also additional available information about the environment into a single SLAM framework. The object oriented design is thought to accommodate different types of information about the environment as long as there is a front-end that can provide this information and a function that can model it. Added information could be in the form of structural, geometric, kinematic, dynamic or even semantic constraints.

The framework consists of: 1) a simulation component that can reproduce several dynamic environments; 2) a front-end that generates the data for the SLAM problem by tracking features, objects and providing point associations using simulated or real data inputs; 3) a back-end component that includes different non-linear solvers for batch and incremental processing. The estimation is implemented as a solution to an NLS problem as presented in section III and solved using Levenberg-Marquardt method. The proposed technique can be easily integrated into any of the existing SLAM back-ends [30], [38], [40]. The code for SLAM



(a) 3 camera rig (b) camera visibility (c) 4 camera rig (d) camera visibility

Fig. 5: Top view of an object (green ellipsoid) circumnavigating a rig of n-cameras counter-clockwise and their visibility maps over time. Each camera and its corresponding visibility map are shown using the same colour. If at a certain time step, more than one camera can see the same object, overlap is present.(best viewed in colour)

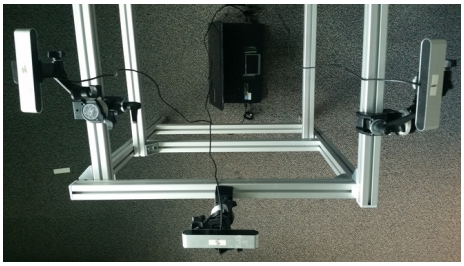


Fig. 6: A snapshot of the real camera calibration data collection; 3 cameras were mounted on a rig, pointing outwards and a checker board was manually moved around the rig, and seen by individual cameras at different time steps.

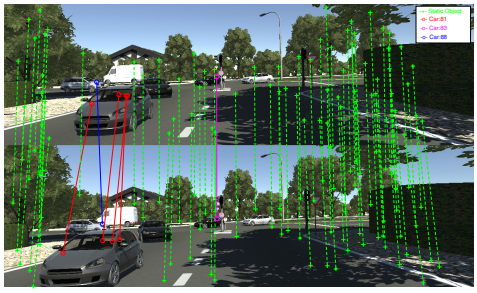


Fig. 7: Feature extraction and tracking applied to the virtual kitti dataset. Static points are shown in green, and a different colour is used for points attached on each unique object.

with dynamic objects will be made publicly available upon acceptance.

V. Experimental results

This section evaluates the proposed technique on the applications described in section IV. We are focused on analysing the accuracy and consistency of the proposed estimation solution and comparing it to the classical SLAM formulation that does not integrate any additional information about the motion of the 3D points in the environment.

The accuracy of the solution of the SLAM problem is evaluated by comparing the absolute trajectory translational error (ATE), the absolute trajectory rotational error (ARE), the absolute structure error (ASE), the all to all relative trajectory translational error (allRTE), the all-to-all relative trajectory rotational error (allRRE), and the all-to-all relative structure error (allRSE) calculated as in [40].

	Exp.A			Exp.B		
Error	**w/o DOM**	**w/ DOM**	**%**	**w/o DOM**	**w/ DOM**	**%**
ATE (m)	0.342	**0.203**	40.6	0.453	**0.351**	22.5
ARE (°)	6.211	**3.751**	39.6	6.371	**4.644**	27.1
ASE (m)	0.733	**0.498**	32.1	0.567	**0.319**	43.7
allRTE (m)	0.213	**0.171**	19.7	0.393	**0.236**	39.9
allRRE (°)	5.183	**3.665**	29.3	5.410	**5.012**	7.3
allRSE (m)	1.049	**0.707**	32.6	0.797	**0.439**	44.9

TABLE I: Error values for experiments 'A' and 'B' explained in section IV-A.1. 'w/ DOM' denotes the proposed estimation technique with dynamic object motion, while 'w/o DOM' means that no object motion information was used in the estimation. A positive % shows improvement of 'w/ DOM'.

A. Analysis of the simulated experiments

The tests show that the proposed method helps preserve the consistency of the map and improves the estimation quality significantly. This can be seen in Fig. 8 and in Table I to Table III.

1) Experiments 'A' & 'B': Table I shows the accuracy results for the Experiments 'A' & 'B'. The values indicate that adding information about the motion of the objects in the estimation process significantly improves the estimation quality and reduces the trajectory and map errors in both absolute and relative metrics. This can be seen in the positive percentage values in the table. Qualitative evaluation of the same experiments is shown in Fig. 4. Note that the ground truth (in green) and the SLAM accounting for constant motion (in blue) have similar values, while without accounting for the constant motion (in red) diverges from the ground truth.

2) "Simulated camera calibration": Results for the multi-camera rig consisting of 3 & 4 cameras are shown in Table II. Results show cm/mm range position accuracy (in the 3 camera calibration and the 4 camera calibration respectively) and less than 1° rotation error.

3) Analysis of the real camera calibration experiment: We project our SLAM 3D point estimates onto their respective camera poses (generated by our SLAM algorithm) to obtain

Error	3 camera calibration	4 camera calibration
ATE (m)	0.098	0.002
ARE (°)	0.265	0.071
ASE (m)	0.041	0.005
allRTE (m)	0.138	0.003
allRRE (°)	0.375	0.007
allRSE (m)	0.223	0.059

TABLE II: Result values for the simulated camera calibration with 3 and 4 cameras, and the real camera calibration dataset.

the structure points in the camera frames. As no ground-truth data is available, a 3D error metric is used. We compare the points in the camera frames to the output of the triangulate function in MATLAB. This triangulate function projects image points onto the camera frames using the knowledge of the baseline between the stereo-cameras. The structure error compared the MATLAB output is 0.01m for the absolute structure error (ASE) and 0.015m for the all-to-all relative structure error (allRSE).

4) Experiment "vKITTI": Table III shows the accuracy of the SLAM solution obtained with and without constant motion information. As expected, the solution of the estimation improves when accounting for moving objects in the scene. It can also be seen in Fig. 8 where a colour map is used to evaluate the error in the returned map. The lighter the colour of the point, the less error it has compared to ground-truth, and the darker the point, the higher the error. Results show a drop of map error from $5.043m$ to only $0.779m$ when accounting for rigid bodies motion.

	vKITTI Exp.		
Error	**w/o DOM**	**w/ DOM**	**%**
ATE (m)	1.815	**1.547**	14.7
ARE (°)	111.624	**8.322**	92.5
ASE (m)	5.043	**0.779**	84.5
allRTE (m)	1.189	**1.082**	8.9
allRRE (°)	77.287	**7.123**	90.7
allRSE (m)	7.325	**1.123**	84.6

TABLE III: Result values for the vKITTI experiment. "w/ DOM" denotes the proposed estimation technique with object motion, while "w/o" means that no object motion information was used in the estimation.

VI. Conclusion and Future Work

In this paper, we introduce a new way to incorporate motion of rigid bodies into a SLAM framework and show how such information can be useful for the SLAM estimation to improve accuracy and consistency of the results. The formulation has the advantage that no additional object pose or knowledge about the object geometry is required in order to account for the moving objects in SLAM. We show potential applications of the proposed algorithm for SLAM in urban environments and for the extrinsic calibration of a multi RGBD camera system. Results show improvements in the estimation quality and consistency of the results compared to the same problem with no added motion information.

Although the calibration examples presented in this paper only use RGBD cameras, the proposed formulation can easily be extended to monocular cameras, to achieve extrinsic and intrinsic camera calibration by minimising re-projection errors. We plan to extend our formulation in the future to include monocular cameras intrinsic, and extrinsic calibration.

Another important issue to be analysed in the future is the computational complexity of SLAM with dynamic objects. It is important to note that without further reductions, the problem can become intractable in large-scale environments with many moving objects. But at the same time, state reduction can be easily implemented using a windowing approach that maintains only a small set of object points in the state rather than the full set of points observed along the entire trajectory. A principled way to do so is to analyse how much the old observations contribute to the solution of the SLAM problem [38], [39], [42]. In the future, we plan to restrict the optimisation problem to relevant state space, and produce scalable solutions.

Acknowledgments

This research was supported by the Australian Research Council through the "Australian Centre of Excellence for Robotic Vision" CE140100016.

References

[1] A. Walcott-Bryant, M. Kaess, H. Johannsson, and J. J. Leonard, "Dynamic pose graph slam: Long-term mapping in low dynamic environments," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2012, pp. 1871–1878.

[2] D. Hahnel, D. Schulz, and W. Burgard, "Map building with mobile robots in populated environments," in *IEEE/RSJ International Conference on Intelligent Robots and Systems, 2002.*, vol. 1. IEEE, 2002, pp. 496–501.

[3] D. Hahnel, R. Triebel, W. Burgard, and S. Thrun, "Map building with mobile robots in dynamic environments," in *IEEE International Conference on Robotics and Automation, 2003. Proceedings. ICRA'03*, vol. 2. IEEE, 2003, pp. 1557–1563.

[4] D. F. Wolf and G. S. Sukhatme, "Mobile robot simultaneous localization and mapping in dynamic environments," *Autonomous Robots*, vol. 19, no. 1, pp. 53–65, 2005.

[5] H. Zhao, M. Chiba, R. Shibasaki, X. Shao, J. Cui, and H. Zha, "Slam in a dynamic large outdoor environment using a laser scanner," in *IEEE International Conference on Robotics and Automation, 2008. ICRA 2008.* IEEE, 2008, pp. 1455–1462.

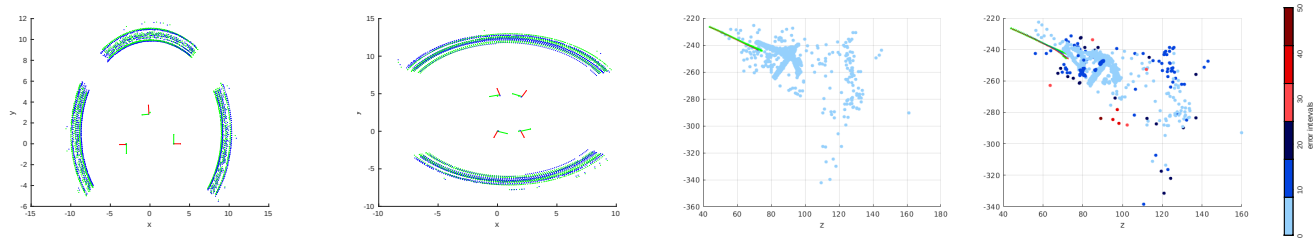
[6] C.-C. Wang, C. Thorpe, and S. Thrun, "Online simultaneous localization and mapping with detection and tracking of moving objects: Theory and results from a ground vehicle in crowded urban areas," in *IEEE International Conference on Robotics and Automation, 2003. Proceedings. ICRA'03*, vol. 1. IEEE, 2003, pp. 842–849.

[7] I. Miller and M. Campbell, "Rao-blackwellized particle filtering for mapping dynamic environments," in *IEEE International Conference on Robotics and Automation, 2007*. IEEE, 2007, pp. 3862–3869.

[8] J. G. Rogers, A. J. Trevor, C. Nieto-Granda, and H. I. Christensen, "Slam with expectation maximization for moveable object tracking," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2010*. IEEE, 2010, pp. 2077–2082.

[9] W. Tan, H. Liu, Z. Dong, G. Zhang, and H. Bao, "Robust monocular slam in dynamic environments," in *IEEE International Symposium on Mixed and Augmented Reality (ISMAR), 2013*. IEEE, 2013, pp. 209–218.

[10] E. Zamora and W. Yu, "Recent advances on simultaneous localization and mapping for mobile robots," *IETE Technical Review*, vol. 30, no. 6, pp. 490–496, 2013.



(a) simulated 3 cameras calibration with no overlap

(b) simulated 4 cameras calibration with overlap

(c) vKitti final estimates with motion integrated

(d) vKitti final estimates with no motion integrated

Fig. 8: Solution of simulated camera calibration (left) and SLAM in urban environment (right). Green represents ground truth camera poses and points, final pose estimates are drawn as coordinates. In the right, a color map is supplied to evaluate the error in the structure points, the lighter the point, the less error it has vs ground-truth, the darker, the higher error. (best viewed in colour)

[11] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard, "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Transactions on Robotics*, vol. 32, no. 6, pp. 1309–1332, 2016.

[12] P. de la Puente and D. Rodríguez-Losada, "Feature based graph-slam in structured environments," *Autonomous Robots*, vol. 37, no. 3, pp. 243–260, 2014.

[13] M. Kaess, "Simultaneous localization and mapping with infinite planes," in *IEEE International Conference on Robotics and Automation (ICRA), 2015*. IEEE, 2015, pp. 4605–4611.

[14] M. Henein, M. Abello, V. Ila, , and R. Mahony, "Exploring the effect of meta-structural information on the global consistency of slam," in *IEEE/RSJ International Conference on Intelligent Robots and Systems 2017*. The Australian National University, 2017.

[15] M. Hsiao, E. Westman, G. Zhang, and M. Kaess, "Keyframe-based dense planar slam," in *IEEE International Conference on Robotics and Automation (ICRA), 2017*. IEEE, 2017, pp. 5110–5117.

[16] B. Mu, S.-Y. Liu, L. Paull, J. Leonard, and J. P. How, "Slam with objects using a nonparametric pose graph," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2016*. IEEE, 2016, pp. 4602–4609.

[17] R. F. Salas-Moreno, R. A. Newcombe, H. Strasdat, P. H. Kelly, and A. J. Davison, "Slam++: Simultaneous localisation and mapping at the level of objects," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2013*. IEEE, 2013, pp. 1352–1359.

[18] C.-C. Wang, C. Thorpe, S. Thrun, M. Hebert, and H. Durrant-Whyte, "Simultaneous localization, mapping and moving object tracking," *The International Journal of Robotics Research*, vol. 26, no. 9, pp. 889–916, 2007.

[19] D. Gálvez-López, M. Salas, J. D. Tardós, and J. Montiel, "Real-time monocular object slam," *Robotics and Autonomous Systems*, vol. 75, pp. 435–449, 2016.

[20] R. Girshick, I. Radosavovic, G. Gkioxari, P. Dollár, and K. He, "Detectron," https://github.com/facebookresearch/detectron, 2018.

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

[22] A. Byravan and D. Fox, "Se3-nets: Learning rigid body motion using deep neural networks," in *IEEE International Conference on Robotics and Automation (ICRA), 2017*. IEEE, 2017, pp. 173–180.

[23] P. Wohlhart and V. Lepetit, "Learning descriptors for object recognition and 3d pose estimation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3109–3118.

[24] P. Cheeseman, R. Smith, and M. Self, "A stochastic map for uncertain spatial relationships," in *4th International Symposium on Robotic Research*, 1987, pp. 467–474.

[25] J. J. Leonard, H. F. Durrant-Whyte, and I. J. Cox, "Dynamic map building for an autonomous mobile robot," *The International Journal of Robotics Research*, vol. 11, no. 4, pp. 286–298, 1992.

[26] S. J. Julier and J. K. Uhlmann, "A counter example to the theory of simultaneous localization and map building," in *IEEE International Conference on Robotics and Automation (ICRA), Proceedings 2001*, vol. 4. IEEE, 2001, pp. 4238–4243.

[27] F. Lu and E. Milios, "Globally consistent range scan alignment for environment mapping," *Autonomous robots*, vol. 4, no. 4, pp. 333–349, 1997.

[28] G. Grisetti, R. Kummerle, C. Stachniss, and W. Burgard, "A tutorial on graph-based slam," *IEEE Intelligent Transportation Systems Magazine*, vol. 2, no. 4, pp. 31–43, 2010.

[29] G. Grisetti, C. Stachniss, S. Grzonka, and W. Burgard, "A tree parameterization for efficiently computing maximum likelihood maps using gradient descent." in *Robotics: Science and Systems*, 2007, pp. 27–30.

[30] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "g 2 o: A general framework for graph optimization," in *IEEE International Conference on Robotics and Automation (ICRA), 2011*. IEEE, 2011, pp. 3607–3613.

[31] V. Ila, L. Polok, M. Šolony, P. Smrž, and P. Zemčík, "Fast covariance recovery in incremental nonlinear least square solvers," May 2015, pp. 4636–4643.

[32] C. Bibby and I. Reid, "Simultaneous localisation and mapping in dynamic environments (slamide) with reversible data association," in *Proceedings of Robotics: Science and Systems*, 2007.

[33] F. Dellaert and M. Kaess, "Square root sam: Simultaneous localization and mapping via square root information smoothing," *The International Journal of Robotics Research*, vol. 25, no. 12, pp. 1181–1203, 2006.

[34] G. S. Chirikjian, R. Mahony, S. Ruan, and J. Trumpf, "Pose changes from a different point of view," in *Proceedings of the ASME International Design Engineering Technical Conferences (IDETC) 2017*. ASME, 2017.

[35] L. Polok, M. Solony, V. Ila, P. Smrz, and P. Zemcik, "Efficient implementation for block matrix operations for nonlinear least squares problems in robotic applications," in *IEEE International Conference on Robotics and Automation (ICRA), 2013*. IEEE, 2013, pp. 2263–2269.

[36] M. Kaess, A. Ranganathan, and F. Dellaert, "isam: Incremental smoothing and mapping," *IEEE Transactions on Robotics*, vol. 24, no. 6, pp. 1365–1378, 2008.

[37] S. Agarwal, K. Mierle, and Others, "Ceres solver," http://ceres-solver.org.

[38] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. J. Leonard, and F. Dellaert, "isam2: Incremental smoothing and mapping using the bayes tree," *The International Journal of Robotics Research*, p. 0278364911430419, 2011.

[39] L. Polok, V. Ila, M. Solony, P. Smrz, and P. Zemcik, "Incremental block cholesky factorization for nonlinear least squares in robotics," in *Proceedings of Robotics: Science and Systems*, Berlin, Germany, June 2013.

[40] V. Ila, L. Polok, M. Šolony, and P. Svoboda, "SLAM++-A highly efficient and temporally scalable incremental SLAM framework," *International Journal of Robotics Research*, vol. Online First, no. 0, pp. 1–21, 2017.

[41] A. Gaidon, Q. Wang, Y. Cabon, and E. Vig, "Virtual worlds as proxy for multi-object tracking analysis," in *CVPR*, 2016.

[42] L. Polok, V. Lui, V. Ila, T. Drummond, and R. Mahony, "The effect of different parameterisations in incremental structure from motion," in *Australasian Conference on Robotics and Automation, 2015*. The Australian National University, 2015, pp. 1–9.