A Exploration of Algorithms Enabling a Semantics-Aware Class-Based Probabilistic Dynamic SLAM

by

Samuel Bateman

A thesis submitted to the
Faculty of the Engineering School of the
University of Colorado in partial fulfillment
of the requirements for the degree of
Bachelors of Science
Department of Computer Science
2020

This thesis entitled:
A Exploration of Algorithms Enabling a Semantics-Aware Class-Based Probabilistic Dynamic SLAM
written by Samuel Bateman
has been approved for the Department of Computer Science

DocuSigned by:
Christoffer Heckman
9FABB79A0D3F4EB...
Assistant Professor Christoffer Heckman

DocuSigned by:
Bradley Hayes
87C4BA9136B949D...
Assistant Professor Bradley Hayes

DocuSigned by:
Sina Aghli
50419099829943C...
Dr. Sina Aghli

Date 5/4/2020

The final copy of this thesis has been examined by the signatories, and we find that both the content and the form meet acceptable presentation standards of scholarly work in the above mentioned discipline.

Bateman, Samuel (B.S., Computer Science)

A Exploration of Algorithms Enabling a Semantics-Aware Class-Based Probabilistic Dynamic SLAM

Thesis directed by Assistant Professor Christoffer Heckman

Effective Localization and Mapping is the cornerstone of the recent expansion of robotics research, promising robust state estimation of both the environment and robot systems utilizing Simultaneous Localization and Mapping (SLAM) Algorithms, a probabilistic approach to state estimation. However, despite the promising future these algorithms portray, they are still very brittle under real world uncertainty. A major failure mode of modern SLAM approaches is a reliance on the so called "static world" assumption, stating that all landmarks in the world can be expected to remain static for the lifetime of the robot. However, some of the applications which offer the greatest promise from the deployment of autonomous systems, such as driverless cars, disaster relief systems and mixed human-robot interaction deployments, categorically defy any static assumption. Recent work has introduced the notion of estimating persistence probability of landmarks in a map and removing them below a certain threshold, providing a path to maintain the normally static maps of SLAM over lifelong deployments in dynamic and semi-static environments. This work presents a new formulation which utilizes class-based groups of features, called cliques, to robustly detect map changes without necessarily removing landmarks of partially occluded objects. Further, this work introduces a framework to learn class-based spatio-temporal survival time priors of said cliques online utilizing recent advancements in Object Detection and older contributions to the long standing field of Survival Analysis. The algorithms are then verified in 3D Simulations.

Dedication

To my girlfriend and parents who invest way too much time and effort in my success.

Acknowledgements

Thank you Dr. Heckman for being a fantastic research advisor and friend, consistently guiding my hand while pushing me to succeed on my own. Your time investment and availability for me has been instrumental in my growth over the last couple years, allowing me to contribute to the vast and amazing world of robotics. Thank you also to Dr. Curry for exposing me to the world of academia, making me believe that I can be successful in academic endeavours and being a consistent mentor and friend. Thank you to my committee for analyzing this document as well as listening to my defense presentation.

Contents

Chapter

1 Introduction **1**

1.1 Related Publications 3

2 Review of the Literature **4**

2.1 Simultaneous Localization and Mapping 4

2.2 Object Detection and Tracking 5

2.3 Dynamic and Semantic SLAM 7

3 Joint Clique Persistence Filter **11**

3.1 Clique Persistence 12

3.2 Joint Detection Likelihood 14

3.3 Joint Evidence 15

3.4 False Negative Suppression 16

4 Class-Based Survival Priors using Multi-Agent Tracking with Reidentification **19**

4.1 Multi-Object Tracking and Reidentification 20

4.2 Extracting Clique Survival Priors 22

5 Results and Discussion **26**

5.1 Joint Clique Filter 26

5.1.1 Simulations 26

5.1.2 Results 28
5.1.3 Discussion 30
5.2 Class-Based Survival Priors using Multi-Agent Tracking with Reidentification 32
5.2.1 Preliminary Results 32

6 Conclusion and Future Work **35**

7 Future Work **36**
7.1 Validation on Real Data 36
7.2 Alternate Survival Function Estimators 36
7.3 Going Beyond Instance Segmentation 37

Bibliography **38**

Chapter 1

Introduction

Despite extensive research and development into the field of robotics and autonomous systems, if you ask random person about the robots they interact with on a daily basis, you will likely be met with a blank stare followed with the phrase: "What Robots?" Robots have become pervasive and integrated extensively into industrial applications, automating portions of factory processes and improving productive capacity while lowering costs. However, all of the success of these deployments have been the result of rigid prior expectations as to the environment of the robot: The robot will operate in a confined space undisturbed and people will only enter this space with very constrained behavior or risk injury as the robot has little to no awareness of its environment. Any situation or task outside of this assumed perfect world rapidly breaks down the capabilities of the platform. However, a long touted promise of the field of robotics and autonomous vehicles has been that robots will be deployed in the areas which we operate in every day, which is in stark contrast to the current capabilities.

To be able to handle environments which are not known prior to deployment or are not specifically engineered for robot operation, perceptive capabilities have had to be developed. One of the most important tasks which humans and animals both perform continuously and highly efficiently is that of localization and mapping, that is to say, developing a mental model of the environment surrounding us and locating ourselves relative to that environment. This deceptively simple subconscious task is the cornerstone of our ability to perform navigation

on small and large spacial scales, as well as our ability to perform controlled manipulation tasks. While humans use a collection of mental models and heuristics on different spacial and temporal scales while extracting semantic landmarks (such as buildings of interest or street signs) to handle the uncertainty inherent in the localization and mapping problems and successfully handle these problems, these solutions are largely non-starters for computers which have no capability to build such a sparse semantic landmark based map and still precisely localize and perform precision control actions. Localization and mapping are dual problems in many (but not all) formulations and as such are often solved simultaneously. Algorithms of this style are denoted Simultaneous Localization and Mapping (SLAM).

SLAM algorithms generally utilize the geometric consistency of static environments to estimate relative translation and rotation (relative pose) of a sensor from the last recorded frame. Then, landmarks which are covisible with a previous frame are measured against their expected position and then these differences are integrated into a probabilistic model. The landmark positions and sensor pose is then optimized to produce a Maximum a Priori (MAP) estimate of the true state of the map and sensor [1]. Given certain assumptions are true, this is a effective blueprint to estimate robot and environment state in uncertain environments.

However, the current formulations of SLAM suffer from a few core failings [2] which result in a lack of widespread adoption, one of which is a failure to handle dynamic environments. In general, the most interesting places to deploy robots are those which violate the so called "static-world assumption" underlying the vast majority of SLAM approaches. All human populated environments are uniformly time varying, just on widely differing time scales. For example, buildings and structural environments can be expected to remain mostly consistent on a day to day timeframe, but over multiple years, change due to construction or reorganization can be expected. However, parked cars and chairs, traditionally thought to be "semi-static" objects can be expected to be static over short time horizons, but categorically should be expected to be inconsistent over even hourly time horizons during the middle

of the day. By this argument, the static world assumption is not only invalid but reality is exactly the opposite: every landmark should be expected to be dynamic or semi-static. Thus, it would be reasonable to introduce a methodology for landmark-based SLAM systems to remove landmarks and observations from a probabilistic model to maintain a map which is locally accurate in time. Sufficient reason to remove a landmark could consist of a low probability of continued persistence based on observations and any prior information on the landmark. The works this thesis is built on aim to utilize recent advancements in Object Detection [3] and Convolutional Neural Networks [4] to create groups of nearly rigid landmarks called cliques and robustly predict the persistence probability of these cliques over lifelong SLAM deployments using both observational information as well as class-based priors.

1.1 Related Publications

The work described this thesis resulted in the following In-Progress and Submitted publications. As such, nearly all of the content and figures of these works or working copies of these works have been reproduced here in their entirety.

S. Bateman, K. Harlow, and C. Heckman, **Better Together: Online Probabilistic Clique Change Detection in 3D Landmark-Based Maps**, International Conference on Intelligent Robots and Systems 2020. Submitted.

S. Bateman, K. Harlow, and C. Heckman, **Working Title: Learning Survival Time Priors Online for Probabilistic Change Detection in Visual SLAM**, Conference on Robot Learning 2020. In-Progress.

Chapter 2

Review of the Literature

2.1 Simultaneous Localization and Mapping

The history of Mapping and Localization in Robotics is long and storied. Ever since the introduction of the Extended Kalman Filter (EKF) in the Simultaneous Localization and Mapping (SLAM) problem for Feature-based maps in the 1990s and early 2000s as described in [1] which promised effective, online state estimation of both the robot and its surrounding environment, the field of Probabilistic Robotics has been on a war path to build a robust, scalable, accurate SLAM algorithm to enable a new wave of robots which can handle the uncertainty of environments which humans and animals negotiate with relative ease. The EKF-SLAM approach traditionally uses sparse features detected through some sensor modality which are assigned to landmarks. New landmark states are appended to the state vector of the EKF and then observed landmark states are marginalized by updating a subset of the EKF state using the Schur Complement. Features are assigned to landmarks using techniques such as Maximum Likelihood Data Association [5] and the Joint Compatibility Branch and Bound algorithm [6].

Since that early work, a number of approaches to the SLAM problem have been proposed, sufficient in number that a simple taxonomy of such algorithms is required to speak intelligently on the matter. The primary two classifications of a SLAM algorithm is that of Direct vs. In-Direct and Dense vs. Sparse systems. A SLAM system which uses pixel intensity directly to estimate intra-frame odometry as a optimization problem is said to be

Direct, where as a system which uses repeatably detectable features to observe structured intra-frame similarity is known as Feature-Based or In-Direct. Further, a SLAM system which estimates and models the correlations between all measurements for each measurement is said to be Dense where as a system which models only a subset of those correlations such that a single measurement is only correlated to a select few number of others is known as Sparse. The notion of Sparse or Dense is harking back to use of correlation matrices in a SLAM system, where the matrix would be literally Dense or Sparse [7]. With a sparse correlation, the correlation matrix can be more succinctly described as a Bayesian Network of related measurements. The relation between two measurement distributions is known as a factor. Fittingly, the resulting probabilistic model is known as a Factor Graph, which was pioneered in [8]. Much more information on this topic can be found in [9].

All SLAM algorithms thus far utilize a *Maximum a Posteriori* (MAP) estimate for the core state estimate, and the vast majority of modern SLAM algorithms utilize a Feature-Based system which utilizes a Factor Graph to encode the distributions of states of a set of landmarks, thus being a Sparse, In-Direct method. Examples of such SLAM approaches is as follows: a Sparse, Indirect SLAM would be ORB-SLAM [10], a Sparse, Direct SLAM would be Direct Sparse Odometry [11], and a Dense, Direct SLAM would be Dense Tracking and Mapping [12]. Dense, Indirect methodologies are rare, however a good example would be the original EKF SLAM which will evolve into a dense covarience matrix over time.

A deeper dive into the generic evolution and State of the Art of SLAM methods can be found in [7].

2.2 Object Detection and Tracking

Object Recognition and Detection are core problems of the field of Computer Vision. For much of the field's history, the problem was completely impenetrable, relying on brittle, hand engineered features such as Color Histograms and energy minimization segmentation using Markov Random Fields [13]. Thus, most object tracking systems up until this point

used these low quality detectors for object proposal with minimal cost data association such as in [14].

However, in the past 10 years or so, a major renaissance on approaches to these problems has emerged with the hardware improvements of Graphics Processing Units enabling Deep Neural Networks (DNNs) [15] to solve high-dimensional, non-linear function approximation problems. With the introduction of DNNs into the problem of Object Recognition in [4], which emerged due to the creation of [16], high performance object recognition was suddenly very possible. Will the advent of Convolutional Neural Networks (CNNs), a number of major computer vision problems were dominated by this technology, such as human pose estimation, facial recognition and, of course, object detection. More information on this can be found in [17].

Object detection is characterized by the problem of predicting bounding boxes for all objects within a image. Within the Object Detection CNN literature, there are two primary approaches: the two stage approach and the end-to-end approach. The two stage approach was spearheaded by [18] and [19] which gather a collection of object proposals, and then classify their "objectness" and class by a second CNN which operates on each proposal. The end-to-end approach started with [20] and was later expanded to [21], which aimed to learn the object detection task end to end by the discretization of image space. This is characterized as end-to-end because it is optimized from detection to input image directly, with no non-differentiable layers. A recent, notable contributions to the end-to-end approach include [**?**], which makes the learning problem of object detection simpler by estimating object centers and height and width rather then 4 object bounding box parameters. While small, this change has a noticeable improvement on accuracy and robustness of the system. The two stage approach has historically resulted in higher performance on standard accuracy metrics, where the end-to-end approaches have historically enabled real-time application with acceptable performance due to much faster evaluation.

A very relevant expansion of the two stage approach is that of [3] which extends the

two-stage object detection framework to perform another task: Instance Segmentation. This is a extension of object detection where, given a object detection, the network also predicts a pixelwise segmentation mask for the object in the bounding box. More recent work in this area [23] has attempted to extend this capability to end-to-end detectors, however the pixelwise accuracy of these end-to-end methods still has much to be desired. In unfamiliar environments where class labels may not be available, [24] proposes a algorithm to enable object detection and instance segmentation of unknown dynamic objects specifically for online SLAM systems.

These object detection approaches, while originally targeted for image-based sensors, have since been expanded to include other sensor modalities such as lidar in [25] and [26]. This trend can be expected to continue and be refined over time, meaning that object detection and instance segmentation based methods can be utilized not only for visual sensor systems, but also for lidar and further sensing modalities.

Recently, these object detection networks have become good enough to be used in robust multi-agent tracking systems such as in [27], which also presents a modern object tracking framework.

2.3 Dynamic and Semantic SLAM

The primary failure modes of SLAM are degradation of state estimate over time, false loop closures, sensor degradation and the "static-world" assumption. This review will primarily focus on the problems of state estimate degradation and the static-world assumption. However, there is now active research into each of these problems, all of which can be reviewed in [2] which provides a excellent survey of the problems faced by the SLAM community going forward.

For the most part, SLAM algorithms which utilize semantic information of scenes to improve state estimate and are robust to dynamics is a open problem. The static-world assumption, which states that all observed landmarks can be expected to remain in the

same place indefinitely, makes the mathematics of the probabilistic model underlying SLAM tractable. However, this comes with a few problems. While highly dynamic features measurements, and therefore objects, can be rejected from Feature-Based systems using outlier rejection, small changes in object positions and environmental composition will result in inconsistent measurements. These landmarks are said to be semi-static, and performing state optimization with measurements of these landmarks over time will accrue drift in the state estimate without ever presenting a clear outlier for rejection.

Another problem with the static-world assumption is that the maps produced by SLAM are often used for control and planning purposes. This means that the landmarks of the map must not only reflect robust features in the environment, but also those that persist over time. However, dynamic objects and changes in environment such as construction will leave many landmarks in the map long past their usefulness, and thus result in free space violations which will inhibit planning algorithms to function properly. So far, there have been a few proposals so solve such problems.

Earlier proposals focused on explicitly tracking the objects in a scene to improve state estimation such as in [28]. However, the authors note that joint estimation of the moving objects and the state variables becomes computationally expensive quickly with few objects, and much of the benefits of the method are lost when doing separate estimates against a map and objects and combining them. None the less, the approach of SLAM at the level of Objects was revisited in [29] which relies on structural priors of objects to extend this approach, allowing for detection of moved objects and localization of the camera given that the majority of objects are not moving. However, the authors of this work noted that their approach often lost tracking over short time frames (10 min) likely due to their reliance on ICP to match sub-maps and objects which accrues error over time. This limits the approaches applicability of the method to lifelong deployments.

Another such proposal is that of [30], which suggests using multiple map hypothesis to handle the problem of dynamic and semi-static landmarks. If a object was to move

through a scene and then cease to be visible upon revisiting the scene, then that hypothesis would be considered lower probability and the estimator would switch to a different map to maintain map coherence. However, this creates a problem with spacial complexity growing enormously with map size, and therefore becomes difficult to argue for when applied to lifelong deployments.

A newer approach, in [31], proposes to mask objects detected by a Instance Segmentation Network and perform in-painting of the new image using previous keyframes before feature detection and processing by the SLAM algorithm. By doing this, the approach attempts to reinforce the static-world assumption which the SLAM algorithm likely relies on: If the system never observes a static or semi-static landmark, the map will be effectively static. This is a good idea, however, like most approaches, it has its failure modes. Chief among them being the problem that one of the major offenders of the semi-static landmark class is that of environmental landmarks over lifelong deployments. A few examples of this are construction work observed by autonomous vehicles or buildings which are destroyed over time in a disaster zone. Even though this approach solves the problem of dynamic objects, it cannot sidestep what is clearly a problem: the environments of robots are time varying and dynamic by nature, and no subset of landmarks can be considered to be always reliable forever because there is no reason to believe any single landmark would exist on a long time horizon.

A approach which addresses this persistence issue is in [32], which proposes that each landmark have a Bayesian filter which estimates the persistence probability given observations and expected observations of a landmark. When the persistence probability is low enough, observations for that landmark would be rejected and eventually the landmark would be removed from the map. This serves a dual purpose of rejecting false positives and removing semi-static and dynamic landmarks from the map over time, improving the quality of the local map at any given time and introducing a way to remove landmarks from a factor graph. This idea was expanded upon in [33] which introduces a joint prior persistence probability

and utilizes it to predict landmark persistence given persistence of a "salient" landmark in a semantic clique of landmarks, allowing it to infer the removal of landmarks without the need to observe the removal. This approach handles many of the problems of the approach of [34], however it is very sensitive to occlusion and instability in any given feature persistence probability, making it poorly suited to real world 3D SLAM. This work aims to resolve the problems with [34] and [33] to expand on the exciting promise of Probabilistic, Dynamic SLAM.

Chapter 3

Joint Clique Persistence Filter

We wish to formulate map change detection in relation to a probabilistic SLAM problem formulation; we will begin by introducing the formalities of the SLAM problem. In its simplest form, SLAM gives the robots' pose $\boldsymbol{x}_p \in \mathbb{SE}(3)$ and map landmarks $\boldsymbol{x}_l \in \mathbb{R}^3$ based on measurements Z which can include odometry information from the robot as well as landmark detections, defined by a sensor model. With this information, the goal of maximum *a posteriori* (MAP) estimate SLAM is to maximize the posterior $p(\mathcal{X}|Z)$ where $\mathcal{X} = [\boldsymbol{x}_p, \boldsymbol{x}_l]$. While this is possible given a limited number of measurements, with any reasonably-sized map this would require summing over all possible measurement associations to obtain the full probability distribution, leading to an intractable calculation [35]. As such, in most problems we must restrict possible measurement associations through a data association step as mentioned in Chapter 2. To do this, we form a vector of data association hypotheses J. Ideally, we would develop this vector by considering the most-likely data associations given our measurements Z and calculating $\text{argmax}_J P(J|Z)$. However, matching a sequence of measurements to a series of discrete landmarks, while also accounting for spurious measurements, is a combinatorially hard problem [6]. As such, most SLAM algorithms instead use a known data association technique such as joint compatibility branch-and-bound (JCBB) [6] or maximum likelihood data association [5] based on sensor model-derived landmark similarity metrics. For a survey of data association techniques, see [36]. Thus, the typical SLAM problem with the static-world assumption is $\text{argmax}_{\mathcal{X}} P(\mathcal{X}|J, Z)$ where J is now the vector

of data associations assigning a single measurement to a single landmark.

We extend this formulation by the introduction of landmark cliques, denoted τ, any one of which is a collection of correlated landmarks, e.g. those on the surface of a rigid body. A similar approach was taken by [33], however we extend the formulation and usage of cliques here. We represent the persistence of a clique as a time varying, binary random variable $\theta_\tau^t \in \{0, 1\}$ where $t \in [0, \infty)$ and $t = 0$ is the first time the clique was observed. Ideally, we could jointly calculate our feature association hypothesis J and the existence of a landmark in a clique θ^t given our measurements Z, but we run into the same problems of intractability and spurious measurements calculating $P(\theta^t, J|Z)$ as we did calculating only $P(J|Z)$. Instead, we condition the landmark persistence on our data associations, calculating $\text{argmax}_{\theta^t} P(\theta^t|J)$ by estimating J using a known data association technique and then estimating θ^t. From these results, we can decide whether or not to continue integrating measurements of a particular element of J based on the probability of θ_τ, or introduce a new feature into the map, while removing the old landmark, similar in spirit to the ideas of [33] and [34]. In this way we can augment $\mathcal{X} = [x_p, x_l, \theta^t]$ and jointly estimate $\text{argmax}_{\mathcal{X}} P(\mathcal{X}|J, Z)$ as before.

3.1 Clique Persistence

In [34, 33], it was assumed that the k^{th} feature in a map M has a survival time $T_k \in [0, \infty)$ such that the feature should be expected to no longer persist for $t > T_k$. This connection between survival time and persistence can be formalized as:

$$P(\theta_k^t|J) = P(T_k \geq t_n|J_k^{1:N}), \tag{3.1}$$

where $j_k^{t_i}$ is the random variable describing the state of a detection of a feature $k \in M$ at possible observation time t_i, denoting the sequence of observations of k as:

$$J_k^{1:N} \triangleq \{j_k^{t_i}\}_{i=1}^N. \tag{3.2}$$

However, the formulation presented in these works doesn't account for the fact that, in 3D, most features on a given object are not observable a large part of the time, even when they are within our perceptive field. To address this, we introduce the clique survival time:

$$T_\tau \sim P_{T_\tau}(\cdot), \tag{3.3}$$

where $P_{T_\tau}(\cdot)$ is approximated by a prior distribution over the survival time T_τ, similar to the feature survival time in [34]. To account for the partial observability of landmarks, we make a similar relation to clique survival time T_τ and our previously introduced clique persistence random variable θ_τ^t as was done in Eq. (3.1); that is,

$$P(\theta_\tau^t|J) = P(T_\tau \geq t_n|J_\tau^{1:N}), \tag{3.4}$$

where $J_\tau^{1:N}$ is the shorthand for $J_{k\in\tau}^{1:N}$. This carries with it the assumption that $\forall i \in \tau, T_i = T_\tau$, which simply states that all landmarks within a clique are likely to persist so long as the clique persists. This deceptively small change has a major implication: an individual landmark may be observed for only a few frames of observation, but some non-empty subset of the clique of landmarks will almost always be observable at every timestep of observation. This also assumes that clique persistence is independent of the observations of other cliques. While this may not be strictly true, it is a necessary assumption to maintain tractability of the presented problem, as also identified by [33].

Using Eq. (3.4), we can then employ Bayes' Rule to estimate persistence of a clique at a given time t:

$$P(T_\tau \geq t|J_\tau^{1:N}) = \frac{P(J_\tau^{1:N}|T_\tau \geq t_n)P(T_\tau \geq t)}{P(J_\tau^{1:N})}. \tag{3.5}$$

From here, one only needs to estimate the likelihood and the evidence of the clique to estimate persistence.

3.2 Joint Detection Likelihood

In order to calculate the joint likelihood of observations of a clique, we make the assumption that, for $a, b \in \tau$:

$$P(J_a^{1:N}, J_b^{1:N} | T_\tau \geq t_n) = P(J_a^{1:N} | T_\tau \geq t_n) P(J_b^{1:N} | T_\tau \geq t_n), \tag{3.6}$$

which is to say that a sequence of detections for one landmark a is conditionally independent from each other landmark e.g. b, in the clique for example, given persistence of the clique. With this, the joint clique likelihood can be broken into the product of detection likelihoods of individual landmarks:

$$P(J_\tau^{1:N} | T_\tau \geq t_n) = \prod_{k \in \tau} P(J_k^{1:N} | T_\tau \geq t_n). \tag{3.7}$$

We then assume that, given the persistence of a feature $k \in \tau$, the sequence of detections will be fully Markovian, as we do not incorporate information about the projected trajectory of the sensor. This allows us to define the landmark likelihood as:

$$P(J_k^{1:N} | T_\tau \geq t_n) = \prod_{i=1}^{N} P(j_k^{t_i} | T_\tau \geq t_n), \tag{3.8}$$

which then allows us to fully expand the joint detection likelihood into:

$$\begin{aligned} P(J_\tau^{1:N} | T_\tau \geq t_i) &= \prod_{k \in \tau} P(J_k^{1:N} | T_\tau \geq t_i) \\ &= \prod_{k \in \tau} \prod_{i=1}^{N} P(j_k^{t_i} | T_\tau \geq t). \end{aligned} \tag{3.9}$$

By identifying that the t_N likelihood is in the expression for the t_{N+1} likelihood, we can factor the t_N expression out. The clique likelihood can be defined recursively as:

$$P(J_\tau^{1:N+1} | T_\tau \geq t) = P(J_\tau^{1:N} | T_\tau \geq t_i) \prod_{k \in \tau} P(j_k^{N+1} | T_\tau \geq t) \tag{3.10}$$

requiring only the new measurements acquired at time t_{N+1} to update the clique likelihood.

Finally, we must calculate the individual detection likelihoods $P(j_k|T_\tau)$ in order to fully define the clique likelihood in Eq. 3.5. With a perfect sensor, this would simply be a binary random variable such that:

$$P(j_k|T_\tau) = \begin{cases} 1, & T_\tau \geq t \\ 0, & T_\tau < t, \end{cases} \tag{3.11}$$

in the case where we expect τ to be observable by the sensor. This does not however account for the probability of falsely detecting a non-existent landmark, denoted P_F, or the probability of missing detection for a existing landmark, denoted P_M, which is particularly troublesome in 3D due to occlusion and sensor degradation, e.g. sensor range limitations. Previous works utilized a fixed constant for P_F and P_M, but this does not capture the complexity of real sensors in general. To remedy this, we model sensor degradation by making P_M a function of minimum clique landmark distance from the sensor, denoted $P_M(s)$. It is common for perception systems to have a maximum detection range $s_{\max}$ but for measurements to only be incorporated into a model, such as a SLAM system, up to a smaller threshold s_{obs} where sensor data is more reliable. We use this to define a range-based sensor degradation model $P_M(s)$ as follows:

$$P_M(s) = 1 - \exp\left(-s \cdot s_{\max}/s_{\text{obs}}\right). \tag{3.12}$$

With P_M and P_F defined, we can utilize the formulation of the feature detection likelihood from [34]:

$$P(j_k|T_\tau) = \begin{cases} P_M^{1-j_k^t}(1-P_M)^{j_k^t}, & T_\tau \geq t \\ P_F^{j_k^t}(1-P_F)^{1-j_k^t}, & T_\tau < t. \end{cases} \tag{3.13}$$

3.3 Joint Evidence

The derivation of the Joint Clique Evidence is inspired by [34], where we capitalize on the conditional independence of landmark detection. Because the clique likelihood defined in Eq. (3.7) is only updated at discrete times $\{t_i\}_{i=1}^N$, the likelihood is constant over the

intervals in the set:

$$\{[t_i, t_{i+1}) | \forall i \in \mathbb{Z}, 0 \leq i \leq N\}. \tag{3.14}$$

As in [34], we can simplify our integral by defining $t_0 \triangleq 0$ and $t_{N+1} \triangleq \infty$. Thus we can calculate the evidence over cliques as:

$$\begin{aligned} P(J_\tau^{1:N}) &= \int_0^\infty P(J_\tau | T_\tau) P(T_\tau) dT_\tau \\ &= \sum_{i=0}^{N} \int_{t_i}^{t_{i+1}} P(J_\tau | T_\tau) p(T_\tau) dT_\tau \\ &= \sum_{i=0}^{N} \prod_{k \in \tau} P(J_k^{1:N} | T_\tau) [F_{T_\tau}(t_{N+1}) - F_{T_\tau}(t_N)], \end{aligned} \tag{3.15}$$

where $F_{T_\tau}(t)$ is the cumulative distribution function of $P_{T_\tau}(t)$ which is described in Eq. (3.3). In the spirit of [34], we can break the clique evidence $P(J_\tau^{1:N})$ into a lower partial sum:

$$L(J_\tau^{1:N}) \triangleq \sum_{i=0}^{N-1} \prod_{k \in \tau} P(J_k^{1:N} | T_\tau) [F_{T_\tau}(t_{N+1}) - F_{T_\tau}(t_N)], \tag{3.16}$$

so that we can define the evidence recursively as:

$$P(J_\tau^{1:N}) = L(J_\tau^{1:N}) + \big(\prod_{k \in \tau} P(J_k^{1:N} | T_\tau) \big) [1 - F_{T_\tau}(t_N)]. \tag{3.17}$$

Then $L(J_\tau^{1:N})$ related to $L(J_\tau^{1:N+1})$ as:

$$\begin{aligned} L(J_\tau^{1:N+1}) = &\big(\prod_{k \in \tau} P_F^{j_k^{t_{N+1}}} (1 - P_F^{j_k^{t_{N+1}}}) \big) \\ &\big(L(J_\tau^{1:N}) + P(J_\tau^{1:N} | t_n \geq T_\tau) [F_T(t_{N+1}) - F_T(t_N)] \big), \end{aligned} \tag{3.18}$$

admitting the full recursive Bayesian estimation procedure as outlined in Algorithm 1.

3.4 False Negative Suppression

A major shortcoming of Bayesian landmark filters in 3D is that cliques, and their constituent landmarks, are only partially observable due to sensor degradation, map geometry, and map occlusion. However, in a joint filter formulation such as in [33] and here, the filter

Algorithm 1 Algorithm for recursively calculating Joint Clique Filter (JCF) posterior persistence probability

Input: Feature detector error probabilities (P_M, P_F), cumulative clique prior F_{T_τ}, feature detector outputs $J_\tau^{t_i}$.

Output: Persistence beliefs on clique $P(\theta_\tau = 1|J_\tau^{1:N})$ for $t \in [t_N, \infty)$.

Initialization: Set $t_0 \leftarrow 0$, $N \leftarrow 0$, $P(J_\tau^{1:0}|t_0) \leftarrow 1)$, $L(J_\tau^{1:N}) \leftarrow 0, P(J_\tau^{1:N}) \leftarrow 1$
while $\exists$ new data $J_\tau^{t_{N+1}}$ **do**
  Compute the partial clique evidence $L(J_\tau^{1:N+1})$ using (3.18)
  Compute the clique likelihood $P(J_\tau^{1:N+1}|T_\tau \geq t)$ using (3.10)
  Compute the clique evidence $P(J_\tau^{1:N+1})$ using (3.17)
  $N \leftarrow N + 1$
  Compute the posterior probability $P(T_\tau \geq t|J_\tau^{1:N})$ using (3.5)
end while

must be updated for all landmarks in a clique if any one of the constituent landmark is visible. This works well when the clique is well within sensor range, but becomes problematic when only a few of the landmarks within a clique are possibly observable, such as when the clique is on the boundary of the observable area. The filter would then update, including in a number of negative detections for which there is no corresponding real information, driving the persistence probability down prematurely.

To remedy this, for all landmark filters we study in this work, we introduce a technique to suppress false negatives: Let $d(\boldsymbol{x}_p, \boldsymbol{x}_{l_k})$ be defined as the Euclidean distance between the positional component of $\boldsymbol{x}_p$ and $\boldsymbol{x}_{l_k}$. Let

$$\tau^* = \{k \in \tau | k \in \text{Positive Detections at } t_n\} \tag{3.19}$$

and pick δ to be a positive threshold ratio of Positive Detections to total Expected Detections. If the following 3 statements are true:

1. $\forall k \in \tau, s_{\text{obs}} < d(\boldsymbol{x}_p, \boldsymbol{x}_{l_k})$,
2. $\exists k \in \tau$ st. $d(\boldsymbol{x}_p, \boldsymbol{x}_{l_k}) < s_{\max}$, and
3. $\dfrac{|\tau^*|}{|\tau|} < \delta$,

then we reject detections for all of τ at this time step. While deceptively simple, this can significantly improve the performance of our Joint Clique Filter in 3D, as well as the Joint Feature filter from [33] and even the Independent Feature Filter from [34], which eschews cliques.

Chapter 4

Class-Based Survival Priors using Multi-Agent Tracking with Reidentification

Thus far, this work has relied on the assumption that cliques would be provided to the SLAM algorithm, where all composite landmarks lie on a rigid body. In addition, we relied on the existence of a Survival Function Prior for use in the Joint Clique Persistence Filter to define the Posterior Persistence Probability. In this section, we will argue that dependence on these assumptions is not only valid, but we will outline a set of algorithms to extract cliques from the environment.

A natural place to start looking for landmarks which are related rigidly and highly correlated is landmarks which lie on a single object. As alluded to in the prior work, recent advances in Computer Vision utilize Deep Learning to effectively detect objects of different classes in a image in the Object Detection Problem. Further, the work in the Instance Segmentation Problem detect masks of a given bounding box for the object it is detecting.

When applied to the context of SLAM, on any given observation timestep, we can use the masks afforded by Instance Segmentation networks, such as Mask R-CNN [3], to predict that the collection of landmarks which are observed within a mask are therefore also on the object which was detected.

This can be utilized to motivate the the following tracking algorithm.

4.1 Multi-Object Tracking and Reidentification

Given that a collection of landmarks which lie within the mask of a object are likely also on that object, tracking the objects behavior would be sufficient to infer much of the behavior of the landmarks which reside on the object. One can then to leverage the work of multi-object tracking such as in [27], which indicates that the best way to track multiple objects in a scene is still to use a set of Kalman Filters [1], performing data association between hypothesis and new observations using the Hungarian Algorithm [37] to solve the optimal assignment problem.

In [27], they track objects using a monocular camera and no temporal photogammetry, and thus require a very complex cost function to successfully perform data association. Because of the nature of the SLAM problem, all perceived features have a perceived 3D pose before assignment to a landmark, whether directly from sensor data or computed from the epipolar geometry between the current frame and a relevant keyframe in which the feature of interest is covisible [38]. This means that by taking the mean of the projected returns, depth estimates, and feature positions, we can estimate the centroid of a object relative to our sensor platform and thus relative to the world given a simple observation timestep. This centroid is biased towards the observing sensor because of the partial observability of a object in the cases of lidar and cameras, making this a somewhat uninformed approach. However 3D pose estimation networks such as [39] combined with the scaling of depth sensing devices can allow for calibration of a good approximation of depth of the centroid of a object. The set of 3D centroid proposal positions and their respective class IDs as denoted by the instance detection framework at time t will be called the Proposal Set, P_t.

Like many other multi-object tracking works, such as [27], we will use the notion of Tracklets to describe a single track in our system at a given time. A Tracklet is composed of a filter modeling at least position and velocity to provide insight into the dynamics of the object at a timestep as well as a prediction as to its position when attempting to track

it at a future time step. The Tracklet also includes a Markovian state of being BIRTHED (just initialized), ALIVE (Actively Tracked), LOST (Actively Tracked up until the last few timesteps), IN_WORKING_SET (Likely a static object, candidate to start tracking again), and DEAD (tracklet has been removed from the working set.

We introduce the concept of the Working Set, W_t is the set of objects where the magnitude of the estimated velocity vector is smaller then a given parameter δ, are no longer visible, and the class of the object is sufficiently likely to persist over a useful time frame. The intuition of the Working Set is that objects which are observed to move exceptionally slowly or not at all are likely static and any observed velocity is as a result of noise in measurement. As we want to track static objects over a time horizon much longer then their short term survival in the local visible frame, we stop candidate tracklets from being moved into the DEAD state and move them into the working set for reidentification later. Our Tracklets in particularly use a constant acceleration model Kalman Filter [1] to marginalize observed positions of matched centroids.

To aid in our goal of predicting clique survival priors, we want to use these tracklets to estimate sample survival times of the tracked objects, that being the time to becoming dynamic. To do this, in any given tracklet, after a "warm-up" of the filter (some number of timesteps in the BIRTH state) to give a good sample of positions for the constant acceleration model to converge, we classify the object as dynamic or static based on a class based threshold and the velocity estimate. For all sequences of timesteps which the object is observed to be in the static state, we record that to a vector of observed survival times. The static state is terminated implicitly when the object is no longer visible.

All of this gives us the tools to describe the multi-agent tracker as in [27] with the addition of long-term reidentification which is depicted in Algorithm 2. This also provides a first pass at clique detection. One can add a list of perceived landmarks which fall inside the projected instance segmentation mask/3d bounding box to each tracklet which is assigned that mask at time t_n. With the co-occurrence of the tracklet system and a set of features

related to each tracklet, if there is a problem with incorrect re-identification using only a epsilon-ball weighting, one can utilize feature matching of the features within the tracklet clique against the proposal to further inform the cost function.

4.2 Extracting Clique Survival Priors

Using the aforementioned Multi-Object Tracking Algorithm, one can store inside each tracklet every timestamp that the object transitions from static to dynamic, and back again. By storing the durations between these state transitions, we have effectively extracted sample survival times of a static object from each of these objects, paired to the class which the tracklet was assigned. Over time, DEAD state tracklets will accumulate in the Dead Set for each class, giving us a effective way to collect survival times proportional to the amount of time which our system is actively perceiving the environment. Using this data, we can estimate a class-based survival time prior function.

In survival analysis, fully parametric models are rarely used for estimating survival functions, particularly when varying hazard functions are of interest because selecting a parametric formulation often predisposes properties of the underlying distribution. If these assumptions are valid, it can work effectively, but in our case of trying to model arbitrary survival functions of class-based survival times, such assumptions would greatly weaken the expressiveness of the prior function which has been noted to be required to be as expressive as possible in [34].

The simplest and most classical way to estimate survival function from a collection of sample survival times is the Kaplan-Meier Estimator [40], often used in the medical and actuarial fields to estimate survival times. The Kaplan-Meier Estimator is a non-parametric Maximum Likelihood Estimator which considers the survival function to be a piecewise constant function which is monotone decreasing, given below. However, there is a issue with using such a model: Censoring. In Survival Analysis, a birth event is assumed to be the time when some event started and a death event is the time when a event ended. In the field of

Algorithm 2 Multi-Agent Tracker with Reidentification

```
Initialize Active Tracklet Set A, Working Set W, Dead Set D
while new proposals N at time t_n do
  if |A| == 0 then
    for Each Proposal p in N do
      A ← Tracklet(p)
    end for
  else if |N| == 0 then
    for Each Tracklet r in A do
      r = predict(r, t_n)
    end for
  else
    Create |A| by |A| + |N| cost matrix C
    for Each Tracklet r in A do
      r = predict(r, t_n)
      for i = 0, i < |A| + |N|, i++ do
        if i < |N|, which is equivalent to saying i is a proposal then
          C[r, i] = mahalanobis(pose(P(i)), r) + off_class_cost(class(P(i)), class(r))
        else
          C[r, i] = CostToLost
        end if
      end for
    end for
  end if
  AssignmentPairs ← Hungarian(C)
  for Each Tracklet r in A do
    if AssignmentPairs[r] is a Proposal then
      update(r, proposal(AssignmentPairs[r]))
    end if
    if MissingTimesteps(r) > MaximumLostTimesteps then
      ClassThreshold ← GetClassThreshold(r)
      if |Velocity(r)| < ClassThreshold then
        Insert r in W
      else
        Insert r in D
      end if
    end if
  end for
```

```
    for Each Proposal p in N do
      if p is not in AssignmentPairs then
        BestMatch ← GetMinCostMatchFromW(p)
        if cost(BestMatch) < ReidThreshold then
          Reactivate(BestMatch)
          Update(BestMatch, p)
        else
          A ← NewTracklet(p)
        end if
      end if
    end for
    D ← ObservableUnmatchedTracklets(W)
  end while
```

medicine, while it is assumed that a illness (particularly chronic and long-duration illnesses) starts roughly at diagnosis and this is assumed to be the birth time, often there will be patients at the end of a study which have not died yet. If one estimates the survival function using the end of the period of observation as the death time, the model will chronically underestimate survival time. This phenomena is known as Right Censoring. One can also observe Left Censoring, being a event for which the birth was not observed but the event was observed at some point thereafter. The combination of Right and Left Censoring in one sample is known as Interval Censored. A deeper survey of such topics can be seen in [41] and [42]. The Kaplan-Meier (KM) Estimator assumes that data is uncensored, and as nearly all of our observational data will be interval censored, it is not applicable to this task. In fact, there is no clear way to argue any classic Machine Learning style regression approach will work on such interval censored data. There exists a generalization of the KM Estimator, the Nonparametric Maximum Likelihood Estimator for interval censored data (NPMLE) [43]. While it has some issues with convergence rate and computational complexity, for the purposes of this work, it provides a good first step for the creation of these types of methods.

However, just using all samples likely will not yield useful distributions. For example, data collected in a parking lot will bias the car class survival distribution towards a certain mean survival time which is uncommon outside of said parking lot. To rectify this, we exploit

the largely planar operation of modern robotics to partition the Working set and Dead set into subsets of samples which occur in discrete spacial grid chunks. This can be implemented as a hash map per xy chunks along the two major axis of translation for a terrestrial robot which point to smaller Working sets and Dead Sets.

This solves two problems. One being the implicit conditioning on samples which occur in the same locality, where the number of xy chunks included in generating a survival function model is increased to decrease the variance of the survival function below a upper bound. The other solved problem is the unbounded growth of the number of possible tracklets to be matched to in the Working set at instantiation of a tracklet. While the Working set would grow completely unbounded otherwise, with a locality based searching, the search space is limited to only realistic matches and a significantly smaller search space when operating in lifelong, large scale deployments. This model also allows for the offloading of tracklet subsets of the Working set and Dead set which are not local to the operation of the active robot to a 3rd party server, which then can make the data available to robots operating in the area.

Chapter 5

Results and Discussion

5.1 Joint Clique Filter

5.1.1 Simulations

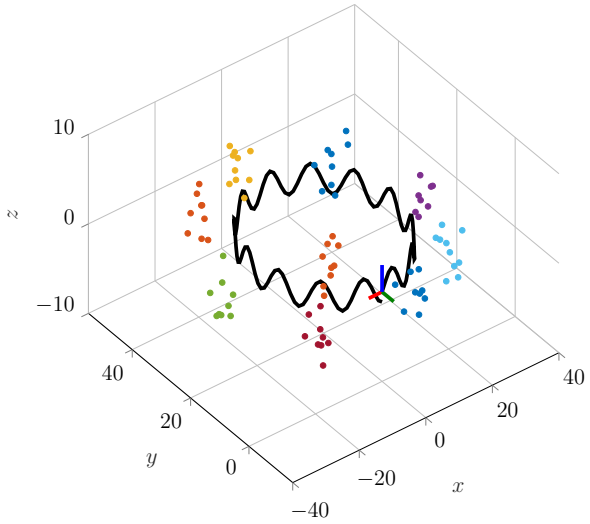

Figure 5.1: Example single-period sensor trajectory simulation with all cliques visible. Most landmarks would not be visible to either of the sensor modalities at the shown time step due to their orientation and the limited field of view of the sensors.

To accurately test the relevant algorithms on realistic 3D data, we developed a 3D simulation of arbitrary, periodic 3D trajectories for a sensor moving past a collection of 3D objects with oriented features corresponding to landmarks, with normals equal to the vector from the center of the object to the feature point. We model two sensor types:

a monocular camera and a 3D lidar. Both sensors have a canonical forward orientation but do not generally move in the forward direction. In simulation we first define $s_{\max}$, the range within which the sensor can detect features, and model detection probability as $P(k) = \exp\left(d(\boldsymbol{x}_p, \boldsymbol{x}_{l_k})/(\lambda)\right)$ with $\lambda = 0.7 \cdot s_{\max}$. For the camera, we use a pinhole projection model where object detection candidates are decided by being within the field of view of the camera oriented along the local x-axis. We model the lidar returns along an annulus around the vertically-aligned coordinates of the sensor frame with a field of view angle above and below the xy plane of the sensor frame.

In our simulations, each object has a uniformly-chosen random number $U \sim [15, 25]$ of features corresponding to landmarks on their surface. Like real landmarks in 3D, corresponding features are not observable from all angles, due to occlusion by the object on which the landmark lies or due to the maximum viewing angle at which a landmark can be detected, distorting their feature-based appearance [10]. This is modeled in simulation by having a maximum angle from the normal of the object at that landmark which is approximated as the vector from the center of the object to the point. If a sensor could detect a landmark that is outside this maximum angle of observation, it will only be detected with probability P_F. The objects stop persisting at a certain time in the simulation, similar to [34], to simulate the movement of a semi-static object or the change in a long term structure.

We run 100 simulations over arbitrary 3D periodic trajectories which pass by the same clique multiple times. An example trajectory and set of cliques is shown in Figure 5.1. All filters are run over the same trajectories for each run to make it as fair as possible. We denote True Positives as TP, False Positives as FP, True Negatives as TN, False Negatives as FN and the total number of possible detection time steps as TR. Our accuracy metric is defined as is traditional as $(TP + TN)/TR$, where a Positive is defined as above and Negative is defined as below a persistence probability threshold ρ_h to allow a measurement to be incorporated to the factor graph.

5.1.2 Results

Sensor Model	Metric	No Negative Suppression				Negative Supression			
		IFF	JFF	JCF	JCFR	IFF	JFF	JCF	JCFR
Lidar	Accuracy	0.705	0.616	0.857	**0.948**	0.734	0.612	0.979	**0.988**
	MES/S	0.185	0.011	0.096	**0.378**	0.259	0.015	0.564	**0.819**
Camera	Accuracy	0.672	0.587	0.793	**0.844**	0.772	0.637	**0.935**	0.910
	MES/S	**0.254**	0.025	0.179	0.170	**0.424**	0.015	0.410	0.293

Table 5.1: IFF: Independent Feature Filter [34], JFF: Joint Feature Filter [33], JCF: Joint Clique Filter (Ours), JCFR: Joint Clique Filter w/ Range Sensor Degradation (Ours). MES/S is Mean Estimated Survival Time over True Survival Time compares the rate at which landmarks where removed to when they should have been removed (Higher is Better). Each of the tested filters attempts to filter changed landmarks from the factor graph. Comparing the JCF/JCFR with the IFF is difficult as the JCF filters integrate a number of measurements at each time step and thus are much more robust to occlusion in 3D environments. The JFF can be directly compared to our methods.

During an example run, we compare various filters by plotting the posterior probability of a given landmark within a clique. In the SLAM problem, a filter will remove that landmark from the map when the posterior drops below a certain threshold ρ_l, and reject new measurements when the posterior probability drops below ρ_h. As shown in Figure 5.2, the Independent Feature Filter clearly removes the given landmark several times, prior to its actual survival time. The Joint Clique Filter with Range Sensor Degradation on the other hand is able to reason over the various missed detections, only removing the landmark from the map after the first observations after the landmarks survival time passes at t=350.

The full results of our simulations can be seen in Table 5.1. The Joint Clique Filter far outperforms the others in accuracy, which is quite important as the Precision of all filters is almost always greater then 99%, meaning that nearly all of the error is coming from False Negatives for each of the filters. A high False Negative count will significantly decrease the number of measurements which can be incorporated into the SLAM algorithm, thereby limiting the accuracy of the map and pose estimation, potentially quite significantly. Based

on the accuracy of our filters, even in a fully static environment, there should be little to no loss of accuracy while incorporating the ability to detect and remove semi-static and dynamic landmarks from a map.

We also define a metric called the Mean Estimated Survival Time over True Survival Time (MES/S) metric which, for a given clique τ, is defined as:

$$\text{MES/S} = \frac{1}{R}\sum_{r=0}^{R}\frac{\hat{T}_\tau^r}{T_\tau^r}, \tag{5.1}$$

where R is the number of runs in a set of simulations, T_τ^r is the actual realization of T_τ in the r^{th} simulation and $\hat{T}_\tau^r$ is the time at which the filter removes the landmark from the map. This metric quantifies efficiency of the landmark filters in a SLAM system as keeping landmarks in a map for the duration of their persistence in the real world. This is discussed in detail in **??**. For lidar, our results are a significant improvement over both Independent Feature Filter (IFF) [34] and the Joint Feature Filter (JFF) [33], where the Joint Clique Filter (JCF) and Joint Clique Filter with Range Sensor Degradation (JCFR) achieve fantastic MES/S with negative suppression and therefore imposing little to no overhead then necessary over the SLAM algorithm to provide feature persistence detection. For cameras, while our method still does well with negative suppression, it is much closer to the Independent Feature Filter in terms of performance under ideal conditions for the IFF. It is important to note, however, that in all of these sensing modalities the IFF can do arbitrarily badly on both the accuracy and MES/S metric given occlusions which commonly occur in 3D environments of interest. This makes comparison of the methods difficult as the IFF lacks the inherent robustness of a joint filter utilizing all the information of the clique, making the JFF the only algorithm that can be compared directly with our work. None the less, the JCF and JCFR still both perform competitively in against IFF in the Camera sensing modality in the MES/S metric, and better in every other metric and sensing modality.

5.1.3 Discussion

When incorporated into a real SLAM pipeline, a Negative detection would result in the removing of a landmark from the factor graph. Thus, a filter which underestimates the true survival time extensively of a landmark would cause excessive operations to be performed on the factor graph and measurement information to be lost. For this reason, we introduced MES/S defined in Equation (5.1), which accounts for the inefficiency of a filter in prematurely removing landmarks from a graph before it has actually changed. As the goal of these filters is to keep landmarks in the map for as long as possible without incurring error, this is a excellent measure of how efficiently they perform the task they are designed to solve when combined with a high accuracy. Ideally, this metric should be as close to 1 as possible, indicating that landmarks are rarely removed from the map prematurely and then added back in afterwards. The higher the value, the less early removals from the map and thus the smaller penalty for performance and accuracy of the SLAM algorithm. A very low MES/S value would indicate constantly removing landmarks and subsequently all measurement constraints from the factor graph, significantly weakening the estimation.

This is a major weakness of the filters from the previous work [33, 34] we compare to here as they often will produce a False Negative very early due to the geometric nature of the features which is not handled in either of these works, pushing this metric very low. To allow these previous filters to perform better on the MFS/S metric, we defined a removal threshold ρ_l which is less then our measurement filter threshold ρ_h as the persistence probability threshold to remove a landmark from the map.

In addition, the tracked landmark in the clique for simulations of the Independent Feature Filter [34] is always observable from the trajectory so that it doesn't unfairly get suppressed by trajectories which cannot detect their corresponding landmark. These modifications do give a unfair edge to the previous methods, however we believe it is better to compare our filter in a more difficult scenario because it makes any positive results stand

much more strongly on their own. In reality, the problem of the observability for the Independent Feature Filter is much more dire. At every observable timestep, nearly all occluded landmarks in a scene will be quickly removed from the map by the Independent Feature Filter, vastly weakening the strength and reusability of the map for any trajectory except for the one being followed by the current sensor. This severely limits the Independent Feature Filter's practical applicability to 3D SLAM workloads, which will not be able to perform loop closure if a large number of landmarks have been removed from the map upon revisiting a location. Our method addresses this major drawback, having the ability to keep landmarks which cannot be observed in the map while still removing dynamic and semi-static landmarks.

The addition of the sensor degradation range prior from the JCF to the JCFR on the lidar appears to be a strict improvement, modeling well the way that lidar detects features and observes objects. This doesn't necessarily seem to be the case in the camera modality though. This is likely because objects can get very close to the camera and then exit the frame, which would denote through the range prior that the filter should nearly always detect the landmarks, but the landmarks would suddenly disappear. This also gets around the negative suppression, explaining why the JCF is dominant with a much smaller margin in this sensing modality, particularly with the JCFR. It is likely that incorporating a notion of the angle of the landmark projection vector from the principle axis of the camera in the sensor degradation model or the negative suppression would resolve this minor dispute.

A interesting note is that all of the discussed filters are highly resilient to large variance in the survival time. This reinforces the belief that observability really is the most important factor in the success of these filters in 3D, giving a large edge to our filter which is the only one which can handle these challenges.

Finally, the False Negative Suppression clearly improves the performance of all filters except for maybe the JFF from [33], which doesn't see much performance improvement from the method. It clearly identifies a major problem of these filters: the fact that all potentially

observable landmarks must be updated at once and landmarks are poorly observable at larger distances and angles of observation. False Negative Suppression provides a simple and satisfying solution to this problem.

5.2 Class-Based Survival Priors using Multi-Agent Tracking with Reidentification

5.2.1 Preliminary Results

While the survival prior extraction algorithm has yet to be deployed on real datasets, preliminary simulations for qualitative observations have been performed. The simulation of tracked objects creates a set of "objects" of different classes which each perform different class based behavior. Each object follows a predetermined acceleration curve with initial velocity and position to produce a trajectory in 3D space. Objects which are within some radius of the "sensor" (the mouse, in this case) then observed with some added Gaussian noise at each measurement.

An example of class-based behavior in one simulation is as follows: one class of objects moves along a random constant velocity trajectory while another has a near zero velocity. In this situation, we observe that the multi-object tracking system is capable of differentiating between static and dynamic objects effectively despite noise and extracts their class-based survival times in real time without issue. This simulation is obviously ideal for this system, however, this proof of concept can be easily applied to real datasets in future work. A screenshot of this simulation can be seen in Figure 5.3

Another example class-based behavior is that of a class of orbiting objects. Because the Kalman Filter depends on a constant acceleration model, it struggles to track the moving orbiting objects as well as the objects moving in a straight line. However, it is unlikely this will matter in a real deployment as static survival time statistics of moving objects is not clearly defined anyway, thus losing tracking and starting tracking on dynamic objects is

rarely of concern as long as the tracklets don't match to static objects in the process.

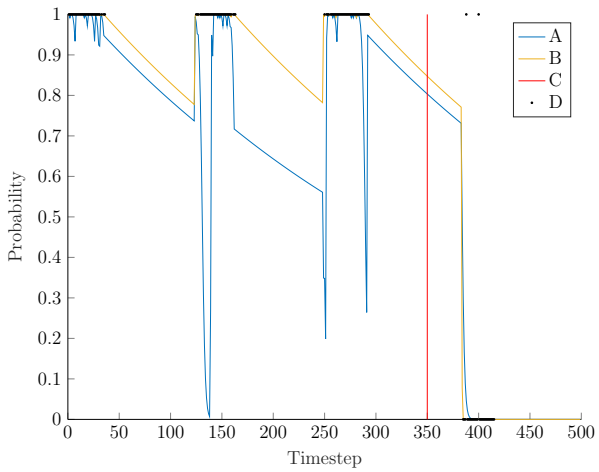

Figure 5.2: Comparison of Example Simulation of the Independent Feature Filter (A) [34] against the Joint Clique Filter with Range Sensor Degradation (B), plotting clique survival time (C) and at least 1 detection (1) or no detections (0) detections from the clique for each expected observation (D). Note that the Independent Feature Filter is far less stable and has many discrepancies with the Joint Clique Filter with Range Sensor Degradation due to the inherent partial observability of landmarks in 3D.

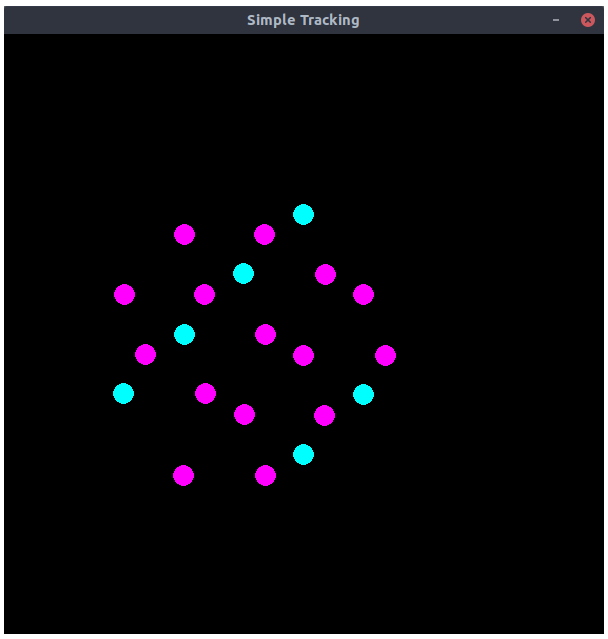

Figure 5.3: Example 3D object tracking simulation frame. All static objects classifications are blue and all dynamic object classifications are purple. Despite a large number of objects as well as many objects in the working set, the object tracker runs at 500+ frames per second, meaning that its overhead on any visual object detection and/or SLAM algorithm is minimal at best, essentially providing survival priors computationally for free.

Chapter 6

Conclusion and Future Work

In the presented works, we have presented a novel formulation of the Bayesian Feature Persistence Filter, called the Joint Clique Filter, which is robust in 3D environments and was validated in a 3D Simulation. Furthermore, we designed a system to extract survival time statistics online from a SLAM system using recent work in Object Detection and Tracking and extended the state of the art tracking algorithms for lifelong deployments. This was then also underwent preliminary testing in a 3D Simulation.

The algorithms presented in this work open the door to deploying a Dynamic, Semantic SLAM algorithm on real data which effectively culls semi-static and dynamic landmarks from a local map online using information from observed features and class-based, locally sensitive priors learned online in a given environment. This gives a first step towards truly lifelong, maintainable SLAM algorithms operating on temporally sensitive maps without the spacial complexity of multiple hypothesis while being resilient to even dynamics and deformation of semantically "static" objects.

Chapter 7

Future Work

7.1 Validation on Real Data

The first avenue for further research and exploration is to validate the multi-agent tracking pipeline against real data, such as [44], and view the resulting survival models. While this wouldn't provide quantitative evaluation, it would better solidify the effectiveness of this approach qualitatively if there are features of interest (such as non-exponential behavior around parking lots for instance) in the resulting survival time priors.

If this appears to be effective, then it would warrant the implementation of a sparse, feature-based SLAM system which incorporates the algorithms outlined in this work and could then be also evaluated on [44] for the quantitative localization performance improvement as well as the qualitative improvement in the long-term viability of the map. This would then test this entire stack of algorithms, building a dynamic, semantic SLAM system.

7.2 Alternate Survival Function Estimators

Another interesting expansion on this work is to utilize different survival function estimators. As noted in Chapter 4, one cannot directly regress to the survival function of interval censored measurements, which is exclusively the survival time measurements which the Multi-Object Tracking with Reidentification can provide. This means that we must be more careful when proposing other possible estimators then the traditional Kaplan-Meier which was suggested in Chapter 4.

Another proven effective estimator for interval censored survival analysis data is the Gaussian Process (GP), as described in [45]. These could be used to estimate a survival function from a distribution accurately and with data efficiency without imposing a predetermined distribution for the hazard function. In addition, the use of a GP in lieu of the Kaplan-Meier estimator allows for quantitative explanatory variables (such as velocity or a approximate size of the object) to be directly used as inputs to the GP and directly effect the resulting probability.

The relatively small number of datapoints per locality means that the normal pitfall of algorithmic complexity in GPs is avoided for the most part, making this a highly desirable alternative.

7.3 Going Beyond Instance Segmentation

One major pitfall of the proposed system is that it doesn't provide semantic priors or cliques to environmental objects which are not classified as "objects" by most instance segmentation nets (such as trains or buildings). In previous works such as [33], cliques were proposed via nearest neighbors, however this doesn't necessarily retain the same semantic information which is available to our method. A simple tweak would be to utilize the result of a Semantic Segmentation, such as one of the results of [3], to classify landmarks which do not fall under a object class (and thus are likely semi-static) and then only create nearest neighbor cliques within a semantic class. This small change permits semantic priors to be used on all landmarks in a map and still have reasonably rigid body cliques because all likely dynamic or non-rigid objects have been classified by the Instance Segmentation network and therefore their landmarks are already assigned to a clique. Further extension of this would entail expanding learned survival priors into this other type of clique which cannot be tracked directly like the Object/Instance Detection approach provided here.

Bibliography

[1] S. Thrun, W. Burgard, D. Fox, et al., "Probabilistic robotics, vol. 1," 2005.

[2] 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 Trans. Robot., vol. 32, pp. 1309–1332, Dec. 2016.

[3] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in Proceedings of the IEEE international conference on computer vision, pp. 2961–2969, 2017.

[4] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," Commun. ACM, vol. 60, pp. 84–90, May 2017.

[5] D. Avitzour, "A maximum likelihood approach to data association," IEEE Transactions on Aerospace and Electronic Systems, vol. 28, pp. 560–566, Apr. 1992.

[6] J. Neira and J. D. Tardos, "Data association in stochastic mapping using the joint compatibility test," IEEE Transactions on Robotics and Automation, vol. 17, pp. 890–897, Dec 2001.

[7] F. Nobre, "A Survey on SLAM for Long-Term Autonomy," p. 35, 2015.

[8] S. Thrun and M. Montemerlo, "The graph slam algorithm with applications to large-scale mapping of urban structures," The International Journal of Robotics Research, vol. 25, no. 5-6, pp. 403–429, 2006.

[9] F. Dellaert and M. Kaess, "Factor Graphs for Robot Perception," FNT in Robotics, vol. 6, no. 1-2, pp. 1–139, 2017.

[10] R. Murartal and J. D. Tardos, "ORB-SLAM2: an Open-Source SLAM System for Monocular, Stereo and RGB-D Cameras," IEEE Trans. Robot., vol. 33, pp. 1255–1262, Oct. 2017.

[11] J. Engel, V. Koltun, and D. Cremers, "Direct Sparse Odometry," arXiv:1607.02565 [cs], Oct. 2016.

[12] 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, pp. 2320–2327, Nov. 2011.

[13] R. Szeliski, Computer vision: algorithms and applications. Springer Science & Business Media, 2010.

[14] L. Zhang, Y. Li, and R. Nevatia, "Global data association for multi-object tracking using network flows," in 2008 IEEE Conference on Computer Vision and Pattern Recognition, pp. 1–8, June 2008.

[15] I. Goodfellow, Y. Bengio, and A. Courville, Deep learning. MIT press, 2016.

[16] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in 2009 IEEE conference on computer vision and pattern recognition, pp. 248–255, Ieee, 2009.

[17] A. Voulodimos, N. Doulamis, A. Doulamis, and E. Protopapadakis, "Deep Learning for Computer Vision: A Brief Review," Computational Intelligence and Neuroscience, vol. 2018, pp. 1–13, 2018.

[18] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," arXiv:1311.2524 [cs], Oct. 2014.

[19] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," arXiv:1506.01497 [cs], Jan. 2016.

[20] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," arXiv:1506.02640 [cs], May 2016.

[21] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," arXiv preprint arXiv:1804.02767, 2018.

[22] Z. Tian, C. Shen, H. Chen, and T. He, "Fcos: Fully convolutional one-stage object detection," in Proceedings of the IEEE International Conference on Computer Vision, pp. 9627–9636, 2019.

[23] D. Bolya, C. Zhou, F. Xiao, and Y. J. Lee, "Yolact++: Better real-time instance segmentation," arXiv preprint arXiv:1912.06218, 2019.

[24] R. Hachiuma, C. Pirchheim, D. Schmalstieg, and H. Saito, "DetectFusion: Detecting and Segmenting Both Known and Unknown Dynamic Objects in Real-time SLAM," arXiv:1907.09127 [cs], July 2019.

[25] Y. Zhou, P. Sun, Y. Zhang, D. Anguelov, J. Gao, T. Ouyang, J. Guo, J. Ngiam, and V. Vasudevan, "End-to-End Multi-View Fusion for 3D Object Detection in LiDAR Point Clouds," arXiv:1910.06528 [cs], Oct. 2019.

[26] C. R. Qi, W. Liu, C. Wu, H. Su, and L. J. Guibas, "Frustum PointNets for 3D Object Detection from RGB-D Data," arXiv:1711.08488 [cs], Apr. 2018.

[27] H.-N. Hu, Q.-Z. Cai, D. Wang, J. Lin, M. Sun, P. Krahenbuhl, T. Darrell, and F. Yu, "Joint monocular 3d vehicle detection and tracking," in Proceedings of the IEEE International Conference on Computer Vision, pp. 5390–5399, 2019.

[28] 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, pp. 889–916, Sept. 2007.

[29] 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 2013 IEEE Conference on Computer Vision and Pattern Recognition, (Portland, OR, USA), pp. 1352–1359, IEEE, June 2013.

[30] T. Morris, F. Dayoub, P. Corke, G. Wyeth, and B. Upcroft, "Multiple map hypotheses for planning and navigating in non-stationary environments," in 2014 IEEE International Conference on Robotics and Automation (ICRA), (Hong Kong, China), pp. 2765–2770, IEEE, May 2014.

[31] B. Bescos, J. M. Fácil, J. Civera, and J. Neira, "DynaSLAM: Tracking, Mapping and Inpainting in Dynamic Scenes," IEEE Robot. Autom. Lett., vol. 3, pp. 4076–4083, Oct. 2018.

[32] D. M. Rosen, J. Mason, and J. J. Leonard, "Towards lifelong feature-based mapping in semi-static environments," in 2016 IEEE International Conference on Robotics and Automation (ICRA), pp. 1063–1070, IEEE, 2016.

[33] F. Nobre, C. Heckman, P. Ozog, R. W. Wolcott, and J. M. Walls, "Online probabilistic change detection in feature-based maps," in 2018 IEEE International Conference on Robotics and Automation (ICRA), pp. 1–9, May 2018.

[34] D. M. Rosen, J. Mason, and J. J. Leonard, "Towards lifelong feature-based mapping in semi-static environments," in 2016 IEEE International Conference on Robotics and Automation (ICRA), pp. 1063–1070, May 2016.

[35] F. Dellaert, Monte Carlo EM for Data-Association and its Applications in Computer Vision. PhD thesis, Carnegie Mellon University, Pittsburgh, PA, September 2001.

[36] A. J. Cooper, "A comparison of data association techniques for Simultaneous Localization and Mapping," Master's thesis, Massachusetts Institute of Technology, United States, 2005.

[37] H. W. Kuhn, "The hungarian method for the assignment problem," Naval research logistics quarterly, vol. 2, no. 1-2, pp. 83–97, 1955.

[38] D. A. Forsyth and J. Ponce, Computer vision: a modern approach. Prentice Hall Professional Technical Reference, 2002.

[39] K. Duan, S. Bai, L. Xie, H. Qi, Q. Huang, and Q. Tian, "Centernet: Keypoint triplets for object detection," in Proceedings of the IEEE International Conference on Computer Vision, pp. 6569–6578, 2019.

[40] E. L. Kaplan and P. Meier, "Nonparametric estimation from incomplete observations," Journal of the American statistical association, vol. 53, no. 282, pp. 457–481, 1958.

[41] P. Wang, V. Tech, Y. Li, and C. K. Reddy, "Machine Learning for Survival Analysis: A Survey," ACM Computing Surveys, vol. 1, no. 1, p. 38, 2017.

[42] D. G. Kleinbaum and M. Klein, Survival Analysis: A Self-Learning Text, Third Edition. Statistics for Biology and Health, New York: Springer-Verlag, 3 ed., 2012.

[43] B. W. Turnbull, "The Empirical Distribution Function with Arbitrarily Grouped, Censored and Truncated Data," Journal of the Royal Statistical Society. Series B (Methodological), vol. 38, no. 3, pp. 290–295, 1976.

[44] W. Maddern, G. Pascoe, C. Linegar, and P. Newman, "1 year, 1000 km: The Oxford RobotCar dataset," The International Journal of Robotics Research, vol. 36, pp. 3–15, Jan. 2017.

[45] T. Fernández, N. Rivera, and Y. W. Teh, "Gaussian processes for survival analysis," in Advances in Neural Information Processing Systems, pp. 5021–5029, 2016.

[46] M. R. Minar and J. Naher, "Recent Advances in Deep Learning: An Overview," arXiv:1807.08169 [cs, stat], 2018.

[47] S. Sabour, N. Frosst, and G. E. Hinton, "Dynamic Routing Between Capsules," arXiv:1710.09829 [cs], Nov. 2017.

[48] F. K. Gustafsson, M. Danelljan, and T. B. Schön, "Evaluating Scalable Bayesian Deep Learning Methods for Robust Computer Vision," arXiv:1906.01620 [cs, stat], Jan. 2020.

[49] P. Billingsley, Probability and Measure. Hoboken, N.J: Wiley, anniversary edition ed., Feb. 2012.

[50] K. Rematas, I. Kemelmacher-Shlizerman, B. Curless, and S. Seitz, "Soccer on Your Tabletop," arXiv:1806.00890 [cs], June 2018.

[51] R. Hu, P. Dollár, K. He, T. Darrell, and R. Girshick, "Learning to Segment Every Thing," arXiv:1711.10370 [cs], Mar. 2018.

[52] K. M. Jatavallabhula, E. Smith, J.-F. Lafleche, C. F. Tsang, A. Rozantsev, W. Chen, T. Xiang, R. Lebaredian, and S. Fidler, "Kaolin: A PyTorch Library for Accelerating 3D Deep Learning Research," arXiv:1911.05063 [cs], Nov. 2019.

[53] K. Wang, F. Gao, and S. Shen, "Real-time Scalable Dense Surfel Mapping," in 2019 International Conference on Robotics and Automation (ICRA), (Montreal, QC, Canada), pp. 6919–6925, IEEE, May 2019.

[54] H. Zhan, C. S. Weerasekera, R. Garg, and I. Reid, "Self-supervised Learning for Single View Depth and Surface Normal Estimation," arXiv:1903.00112 [cs], Feb. 2019.

[55] K. Doherty, D. Fourie, and J. Leonard, "Multimodal Semantic SLAM with Probabilistic Data Association," in 2019 International Conference on Robotics and Automation (ICRA), pp. 2419–2425, May 2019.

[56] Y. Xiang, A. Alahi, and S. Savarese, "Learning to Track: Online Multi-object Tracking by Decision Making," in 2015 IEEE International Conference on Computer Vision (ICCV), (Santiago, Chile), pp. 4705–4713, IEEE, Dec. 2015.

[57] M. Rünz and L. Agapito, "Co-Fusion: Real-time Segmentation, Tracking and Fusion of Multiple Objects," 2017 IEEE International Conference on Robotics and Automation (ICRA), pp. 4471–4478, May 2017.

[58] K. E. Madawy, H. Rashed, A. E. Sallab, O. Nasr, H. Kamel, and S. Yogamani, "RGB and LiDAR fusion based 3D Semantic Segmentation for Autonomous Driving," arXiv:1906.00208 [cs], July 2019.

[59] T. D. Barfoot, State Estimation for Robotics. Cambridge: Cambridge University Press, 2017.

[60] F. Nobre, C. Heckman, P. Ozog, R. W. Wolcott, and J. M. Walls, "Online Probabilistic Change Detection in Feature-Based Maps," in 2018 IEEE International Conference on Robotics and Automation (ICRA), (Brisbane, QLD), pp. 3661–3668, IEEE, May 2018.

[61] N. Atanasov, S. L. Bowman, K. Daniilidis, and G. J. Pappas, "A Unifying View of Geometry, Semantics, and Data Association in SLAM," in Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, (Stockholm, Sweden), pp. 5204–5208, International Joint Conferences on Artificial Intelligence Organization, July 2018.

[62] L. Zhang, L. Wei, P. Shen, W. Wei, G. Zhu, and J. Song, "Semantic SLAM Based on Object Detection and Improved Octomap," IEEE Access, vol. 6, pp. 75545–75559, 2018.

[63] K. Konolige and J. Bowman, "Towards lifelong visual maps," in 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems, (St. Louis, MO, USA), pp. 1156–1163, IEEE, Oct. 2009.

[64] 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, (Vilamoura-Algarve, Portugal), pp. 1871–1878, IEEE, Oct. 2012.

[65] G. D. Tipaldi, D. Meyer-Delius, and W. Burgard, "Lifelong localization in changing environments," The International Journal of Robotics Research, vol. 32, pp. 1662–1678, Dec. 2013.

[66] R. W. Wolcott and R. M. Eustice, "Visual localization within LIDAR maps for automated urban driving," in 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, (Chicago, IL, USA), pp. 176–183, IEEE, Sept. 2014.

[67] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár, "Microsoft COCO: Common Objects in Context," arXiv:1405.0312 [cs], Feb. 2015.

[68] "Factor Graphs and GTSAM," May 2019.

[69] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. Leonard, and F. Dellaert, "iSAM2: Incremental smoothing and mapping with fluid relinearization and incremental variable reordering," in 2011 IEEE International Conference on Robotics and Automation, (Shanghai, China), pp. 3281–3288, IEEE, May 2011.

[70] T. Pohlen, A. Hermans, M. Mathias, and B. Leibe, "Full-Resolution Residual Networks for Semantic Segmentation in Street Scenes," arXiv:1611.08323 [cs], Dec. 2016.

[71] L. von Stumberg, P. Wenzel, Q. Khan, and D. Cremers, "GN-Net: The Gauss-Newton Loss for Multi-Weather Relocalization," arXiv:1904.11932 [cs], Nov. 2019.

[72] A. J. Davison, "Active search for real-time vision," in In Proceedings of the IEEE International Conference on Computer Vision, pp. 66–73, 2005.

[73] M. Carrasco, "Visual attention: The past 25 years," Vision Research, vol. 51, pp. 1484–1525, July 2011.

[74] A. Borji and L. Itti, "State-of-the-Art in Visual Attention Modeling," IEEE Trans. Pattern Anal. Mach. Intell., vol. 35, pp. 185–207, Jan. 2013.

[75] R. Wang, M. Schwörer, and D. Cremers, "Stereo DSO: Large-Scale Direct Sparse Visual Odometry with Stereo Cameras," arXiv:1708.07878 [cs], Aug. 2017.

[76] J. N. Corcoran, On The Simple and Innite Joy of Mathematical Statistics.

[77] L. Carlone and S. Karaman, "Attention and Anticipation in Fast Visual-Inertial Navigation," IEEE Trans. Robot., vol. 35, pp. 1–20, Feb. 2019.

[78] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation," arXiv:1802.02611 [cs], Aug. 2018.

[79] D. Caduff and S. Timpf, "On the assessment of landmark salience for human navigation," Cogn Process, vol. 9, pp. 249–267, Nov. 2008.

[80] J. Redmon and A. Farhadi, "YOLO9000: Better, Faster, Stronger," in 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), (Honolulu, HI), pp. 6517–6525, IEEE, July 2017.

[81] H. Strasdat, C. Stachniss, and W. Burgard, "Which landmark is useful? Learning selection policies for navigation in unknown environments," in 2009 IEEE International Conference on Robotics and Automation, (Kobe), pp. 1410–1415, IEEE, May 2009.

[82] R. Lerner, E. Rivlin, and I. Shimshoni, "Landmark Selection for Task-Oriented Navigation," IEEE Trans. Robot., vol. 23, pp. 494–505, June 2007.

[83] P. Sala, R. Sim, A. Shokoufandeh, and S. Dickinson, "Landmark Selection for Vision-Based Navigation," IEEE Trans. Robot., vol. 22, pp. 334–349, Apr. 2006.

[84] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," arXiv preprint arXiv:1804.02767, 2018.

[85] M. Keller, D. Lefloch, M. Lambers, S. Izadi, T. Weyrich, and A. Kolb, "Real-Time 3D Reconstruction in Dynamic Scenes Using Point-Based Fusion," in 2013 International Conference on 3D Vision, (Seattle, WA, USA), pp. 1–8, IEEE, June 2013.

[86] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "On-Manifold Preintegration for Real-Time Visual-Inertial Odometry," IEEE Trans. Robot., vol. 33, pp. 1–21, Feb. 2017.

[87] D. Acuna, A. Kar, and S. Fidler, "Devil is in the Edges: Learning Semantic Boundaries from Noisy Annotations," arXiv:1904.07934 [cs], June 2019.

[88] T. Roughgarden, "CS261: A Second Course in Algorithms Lecture #5: Minimum-Cost Bipartite Matching," p. 14.

[89] T. Bailey and H. Durrant-Whyte, "Simultaneous localization and mapping (slam): Part ii," IEEE robotics & automation magazine, vol. 13, no. 3, pp. 108–117, 2006.

[90] P. Newman and K. Ho, "SLAM-loop closing with visually salient features," in proceedings of the 2005 IEEE International Conference on Robotics and Automation, pp. 635–642, IEEE, 2005.

[91] D. Zapletal and A. Herout, "Vehicle Re-identification for Automatic Video Traffic Surveillance," in 2016 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), (Las Vegas, NV, USA), pp. 1568–1574, IEEE, June 2016.

[92] K. Wang, G. Pleiss, J. Gardner, S. Tyree, K. Q. Weinberger, and A. G. Wilson, "Exact gaussian processes on a million data points," in Advances in Neural Information Processing Systems, pp. 14622–14632, 2019.

[93] J. Engel, T. Schöps, and D. Cremers, "LSD-SLAM: Large-Scale Direct Monocular SLAM," in Computer Vision – ECCV 2014 (D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, eds.), vol. 8690, pp. 834–849, Cham: Springer International Publishing, 2014.

[94] P. Merriaux, Y. Dupuis, R. Boutteau, P. Vasseur, and X. Savatier, "LiDAR point clouds correction acquired from a moving car based on CAN-bus data," arXiv:1706.05886 [cs], June 2017.

[95] S. R. Sukumar, H. Bozdogan, D. L. Page, A. F. Koschan, and M. A. Abidi, "Sensor Selection Using Information Complexity for Multi-sensor Mobile Robot Localization," in Proceedings 2007 IEEE International Conference on Robotics and Automation, (Rome, Italy), pp. 4158–4163, IEEE, Apr. 2007.

[96] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon, "Bundle Adjustment — A Modern Synthesis," in Vision Algorithms: Theory and Practice (G. Goos, J. Hartmanis, J. van Leeuwen, B. Triggs, A. Zisserman, and R. Szeliski, eds.), vol. 1883, pp. 298–372, Berlin, Heidelberg: Springer Berlin Heidelberg, 2000.

[97] X. Zhou, D. Wang, and P. Krähenbühl, "Objects as Points," arXiv:1904.07850 [cs], Apr. 2019.

[98] J. Neira and J. Tardos, "Data association in stochastic mapping using the joint compatibility test," IEEE Trans. Robot. Automat., vol. 17, pp. 890–897, Dec. 2001.

[99] S. R. Giolo, "Turnbull's nonparametric estimator for interval-censored data," Department of Statistics, Federal University of Paraná, pp. 1–10, 2004.

[100] C. Anderson-Bergman, "icenReg : Regression Models for Interval Censored Data in R," J. Stat. Soft., vol. 81, no. 12, 2017.

[101] L. Tian and R. Olshen, "Survival Analysis: Weeks 2-3," p. 33.

[102] J. Engel, J. Stückler, and D. Cremers, "Large-scale direct SLAM with stereo cameras," in 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 1935–1942, Sept. 2015.

[103] 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, pp. 889–916, Sept. 2007.