Articulation Estimation Using Depth Sensing

Suren Kumar
Mechanical and Aerospace Engineering
State University of New York at Buffalo
Buffalo, NY, USA
Email: surenkum@buffalo.edu

Vikas Dhiman
Electrical Engineering
University of Michigan
Ann Arbor, MI, USA
Email: dhiman@umich.edu

***Abstract*—**

- **Detect distinctly moving clustered points/voxels/objects in a scene.**
  - **Use Kinect fusion to create a static map.**
  - **Use some kind of noise threshold to detect object movement independent of camera movement.**
  - **Trigger algorithm (may be use RANSAC ?? etc.) that will segment out the object that just moved. The object should be spatially clustered and should be explained by the same rigid 3D motion.**
  - **Maintain a pairwise relative localization graph of the scene.**
- **Semantic reasoning in map update of these objects and their localization. Reason about Physical support and articulated linkage.**
- **Build a 3D reconstruction of these objects.**
- **Find similar unmapped static objects in the scene. May be use Jeff's detection and segmentation code.**
- **Try algorithm for long term mapping (a week) by using auto charging turtlebots in a living room and compare with existing algorithms.**

I. Introduction

Imagine a robot moving in a typical living room environment which encounters indoor objects such as doors, drawers and chairs etc. We posit that in order for the robot to understand, map or interact with such objects, the robot needs to be able to understand the articulation. Pyschophysical experiments on human motion understanding have demonstrated that human first distinguish between competing motion models (translation, rotation and expansion) and then estimate the motion conditioned on motion model [1].

For any pose estimation task, it is essential to know the articulated structure in order to represent the pose state. We refer to information about joints such as type of joints (Ex: Revolute), number of joints and kinematic chain as articulated structure. The most common solution to this problem is to "detect" the objects (Ex: People [2]) using sensor data and conditioned on the object detection, the articulation structure is known a prior (Ex: Pose Estimation for Humans [3]). Because of the tremendous improvements in the object detection over large datasets [4], solutions that address the problem of articulation structure have taken a backseat. However, we argue that despite the success in visual detection, object detection in unstructured environment still remains an open ended problem. The performance is further reduced on texture-less objects [5] which populate our indoor environments such as doors, drawers, chairs etc. Furthermore, machine learning based approaches only generalize to the objects in the training dataset which limit the applicability of "detection" first methods to previously unseen objects.

II. Related Work

A. Structure from Motion

Using image motion to understand motion and structure in the scene is a historically well studied problem in computer vision. Ullman [6] proposed that in non-degenerate cases under orthographic projection, three pictures of four points can determine structure and motion. Tomasi and Kanade [7] formalized Ullmans's idea and proposed one of the influential method to compute camera motion and image structure by tracking features in the images. They proposed factorizing a matrix of feature tracks into motion and shape matrix by enforcing the rank constraint of the rigid body motion and metric constraints of a rotation matrix. Costeira and Kanade [8] extended the factorization idea to segment and recover shape along with motion of multiple moving bodies in the scene. The resulting motion of rigid bodies can be further analyzed to estimate kinematic chains [9] and hence to yield articulated structures.

There are certain fundamental limitations to structure from motion approaches. First, the reliance on feature tracking methods such as KLT is not suitable for indoor environments which may not have much texture. Secondly, motion orthogonal to image plane is not modelled [9] because image projection is approximated as affine projection. With the discovery of cheap and commonplace hardware such as Kinect, there is a need to re-examine the traditional structure from motion idea. First, since such hardware already provides depth for a feature point, one already has shape as estimated by traditional structure from motion. Also using depth, one can model the motion orthogonal to image plane. Furthermore, texture-less objects can be tracked better by adding depth edges to the tracking mix. There have been efforts at using depth information by simply using the depth and calibration parameters to directly represent the trajectory in R^3 [10], [11].

B. Direct Motion Sensing Approaches

Another predominant class of methods to estimate articulated structure assumes the motion information of individual parts is directly available. Placement of markers such as ARToolKit [12], checker-board markers, Infrared markers etc. on various parts of the articulated body can yield good estimates of the motion transformation. The placement of markers removes the need of otherwise noisy feature-tracking from the structure estimation process [13], [14], [15], [16]. Another way to get better estimation of articulated motion is via active interaction of robot manipulating an articulated object [17], [16].

In contrast to state-of-the-art methods, we propose performing articulation estimation online. Prior work has relied on collecting data from demonstrations and performing articulation estimation offline. Recently Martin et. al [18] have proposed a framework for online estimation, however there is no explicit probabilistic measure for model confidence to select a articulation model. Our second major contribution is addressing the lack of temporal modelling (Ex:acceleration/deceleration of a door) in articulation estimation. We propose an explicit temporal model for each articulation type which is necessary to make good long-term future predictions. Temporal modelling of arbitrary order allows us to ; i) Track new parts/objects that enter/exit the scene [18], ii) Modelling the entire scene and as a result exploring dependencies between neighbouring objects, iii) Assimilating articulated object motion in Simultaneous Localization and Mapping (SLAM). To the best of our knowledge, this is first work that addresses using articulated objects with arbitrary order temporal models within SLAM.

III. Scene Understanding

Analysis by method such as ours is essential in order to decompose the scene into types of motion that a robot can influence on the scene. For example: Understanding the way a drawer can be opened, fridge door can be opened, what can be moved around in the scene

Other important use cases of motion estimation

- Estimation a joint can induce prior over objects such as revolute joint can induce prior over refrigerator and door
- Visual object identification such as drawer can induce a prior over motion estimation
- Object tracking
- For grasping? – such as how to open a door?

IV. Dynamic World Representation

Real world is dynamic in nature with varying degree of motion such as parking lot which can be assumed to be temporary stationary compared to a road which is always in motion. Previous literature to handle dynamic environments can be divided into two predominant approaches A) Detect moving objects and ignore them, B) Track moving objects as landmarks [19]. In the first approach, using the fact that the conventional SLAM map is highly redundant, the moving landmarks can be removed from the map building process [20]. In contrast, Wang et. al [21] explicitly track moving objects by adding them to the estimation state. However the work assumed that the sensor measurement can be decomposed into observation corresponding to moving and static landmarks which requires good estimate of moving and static landmarks to start with. Furthermore, it was assumed that the measurement of moving object carries no information for the SLAM state estimation implying that the map remains unchanged. A simple counter example is the case of a moving door in an indoor environment which changes the map of the scene.

A. *Known Decomposition of the World*

Object SLAM+ Object Tracking Interacting multiple models

B. *No Prior Information*

In feature based mapping, motion of each feature can be assumed to be independent given the location of the feature at previous time step. In dense mapping, a scene/map be decomposed into n different parts such as chair, door etc. whose shape is known. The parts of the scene $m_k = \{b_k^i\}, 1 \leq i \leq n$ are assumed to move independently and hence the motion of the map can be represented as collection of independent motion of the parts. The true motion model for the each part of the scene is assumed to be one of the motion models $C \in \{C_j\}_{j=1}^{p}$ as represented in Section VI.

V. Articulated Model Representation

We represent all the articulated motion in the world as

$$X(t) = f_M(C, q(t)) \tag{1}$$

where $X(t)$ is the observed motion of an object, $M \in \{M_j\}_{j=1}^{r}$ is one of the r possible motion models, C is the configuration space (Ex: doors open about an axis etc.) and $q(t)$ represents the time-varying motion variables(s) (Ex: length of prismatic joint, angle of door etc.) associated with the motion model M.

This kind of representation wherein non-time varying configuration parameters are separated from time-varying motion variables is beneficial in a multitude of ways. First, it allows for a unified treatment of various types of articulation because of a single and consistent representation of motion variables. Second, this representation can be robustly estimated from experimental data given the reduced number of parameters to be estimated and simultaneous often making the estimation problem linear and hence convex.

A notable omission from our modeling of articulated systems as in Equation 1 is the input to the system such as torque acting on a door, force on a drawer etc. This modeling limitation is due to the passive nature of our sensing approach and not making any other assumption about the agents in the scene. The compensate for the lack of input modeling and still To predict motion at next time step $P(X(t+\delta t)|X(t))$ without modeling the input forces/torques (thus not using a dynamics model), we need to model the propagation of motion variables $P(q(t+\delta t)|q(t))$. Before we proceed to model the temporal evolution of motion variables, we consider the task of configuration estimation.

VI. Articulation Classification

The configuration parameters in Equation 1 are entirely dependent on the type of articulated joint. In this section, we consider the problem of articulation identification from point correspondences over time. Rigid bodies can move in

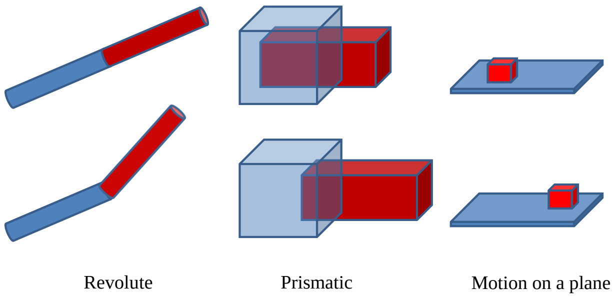


Fig. 1: Some of the articulated joints considered in this work demonstrated at two different time steps. Revolute and prismatic joints are 1 DOF joint while motion on a plane is a 2 DOF joint.

$3D$ space with $SE(3)$ configuration which is product space of $SO(3)$ (Rotation Group for 3D rotation) and $E(3)$ (Translation using 3D movement). The full $SE(3)$ has 6 degrees of freedom (DOF) which is reduced when a rigid body is connected to another rigid body via a joint. For example, the configuration space for a revolute joint which is a 1 DOF joint and can be assumed to be a connected subset of the unit circle. Figure 1 shows some of the articulated joint modelled in this work.

We consider two different types of articulation classification framework: i) Rigid Body Articulation Classification, ii) Single Point Articulation Classification. This distinction is important because a rigid body articulation classification requires observation of at least 3 points on the same body over time. However this approach is not suitable for a variety of use cases where we can only track 1 point on the body or feature based computer vision methods such as Extended Kalman Filter (EKF) SLAM.

A. Rigid Body Articulation Classification

We extend the factorization approach as described in [8] to 3-D track data available from a depth camera. Assuming for now that a single object moves relative to a static camera and we track features from frame to frame. Following the notation in the paper, lets represent a point on the object as $p_i^T = [X_i, Y_i, Z_i]^T$ in the camera frame, in the current frame f, the position of the point in homogeneous coordinates can be represented as

$$s_{fi}^C = \begin{bmatrix} p_{fi}^C \\ 1 \end{bmatrix} = \begin{bmatrix} R_f & t_f \\ 0_{1\times 3} & 1 \end{bmatrix} \begin{bmatrix} p_i \\ 1 \end{bmatrix} = \begin{bmatrix} R_f & t_f \\ 0_{1\times 3} & 1 \end{bmatrix} s_i \tag{2}$$

where R_f and t_f are the rotation and translation of the object from current frame w.r.t to the frame in which object points are initially represented. Assuming that we track N features over F frames, one can write

$$\begin{bmatrix} u_{11} & \dots & u_{1N} \\ . & & . \\ u_{F1} & \dots & u_{FN} \\ v_{11} & \dots & v_{1N} \\ . & & . \\ v_{F1} & \dots & v_{FN} \\ w_{11} & \dots & w_{1N} \\ . & & . \\ w_{F1} & \dots & w_{FN} \end{bmatrix} = \begin{bmatrix} i_1^T & | & t_{x_1} \\ . & | & . \\ i_F^T & | & t_{x_F} \\ j_1^T & | & t_{y_1} \\ . & | & . \\ j_F^T & | & t_{y_F} \\ k_1^T & | & t_{z_1} \\ . & | & . \\ k_F^T & | & t_{z_F} \end{bmatrix} \begin{bmatrix} s_1 & . & . & . & s_N \end{bmatrix} \tag{3}$$

where (u_{fi}, v_{fi}, w_{fi}) is the location of feature point in current frame, vectors i_f^T, j_f^T, k_f^T are the rows of the rotation matrix R_f and $(t_{x_f}, t_{y_f}, t_{z_f})$ represent the components of the translation vector at time instant with frame f. Equation 3 can be represented in a accumulated form as

$$\mathbf{W} = \mathbf{MS} \tag{4}$$

where $\mathbf{W}$ represents the accumulation from trajectories of N points tracked over F frames, $\mathbf{M}$ contains all the information about the motion of the object present in the scene and $\mathbf{S}$ contains all the information about the shape of the object. Since rank of product of two matrices can not exceed the minimum of rank of individual matrics, It is clear that the maximum rank of W is 4. Computing singular value decomposition of $\mathbf{W}$, we get

$$\mathbf{W} = \mathbf{U\Sigma V}^T \tag{5}$$

where $U \in R^{3F\times 4}$, and $V \in R^{N\times 4}$ are left and right real singular matrices and Σ is a 4×4 diagonal matrix of singular values. Although if $\mathbf{W}$ was full rank, we would have to consider N singular values but as the rank of W is 4, we only write out the components corresponding to first 4 singular values. Writing the factorization as product of two matrices,

$$\hat{\mathbf{M}} = \mathbf{U}\Sigma^{\frac{1}{2}}, \hat{\mathbf{S}} = \Sigma^{\frac{1}{2}}\mathbf{V}^T \tag{6}$$

The factorization as defined in Equation 6 is not unique as any invertible 4×4 matrix A will lead to an alternate solution $\mathbf{M} = \hat{\mathbf{M}}A$, $\mathbf{S} = A^{-1}\hat{\mathbf{S}}$

$$\begin{bmatrix} m_{00}^2 & 2m_{00}m_{01} & 2m_{00}m_{02} & m_{01}^2 & 2m_{01}m_{02} & m_{02}^2 \\ m_{10}m_{00} & m_{10}m_{01}+m_{11}m_{00} & m_{10}m_{02}+m_{12}m_{00} & m_{11}m_{01} & m_{11}m_{02}+m_{12}m_{01} & m_{12}m_{02} \\ m_{20}m_{00} & m_{20}m_{01}+m_{21}m_{00} & m_{20}m_{02}+m_{22}m_{00} & m_{21}m_{01} & m_{21}m_{02}+m_{22}m_{01} & m_{22}m_{02} \\ m_{00}m_{10} & m_{00}m_{11}+m_{01}m_{10} & m_{00}m_{12}+m_{02}m_{10} & m_{01}m_{11} & m_{01}m_{12}+m_{02}m_{11} & m_{02}m_{12} \\ m_{10}^2 & 2m_{10}m_{11} & 2m_{10}m_{12} & m_{11}^2 & 2m_{11}m_{12} & m_{12}^2 \\ m_{20}m_{10} & m_{20}m_{11}+m_{21}m_{10} & m_{20}m_{12}+m_{22}m_{10} & m_{21}m_{11} & m_{21}m_{12}+m_{22}m_{11} & m_{22}m_{12} \\ m_{00}m_{20} & m_{00}m_{21}+m_{01}m_{20} & m_{00}m_{22}+m_{02}m_{20} & m_{01}m_{21} & m_{01}m_{22}+m_{02}m_{21} & m_{02}m_{22} \\ m_{10}m_{20} & m_{10}m_{21}+m_{11}m_{20} & m_{10}m_{22}+m_{12}m_{20} & m_{11}m_{21} & m_{11}m_{22}+m_{12}m_{21} & m_{12}m_{22} \\ m_{20}^2 & 2m_{20}m_{21} & 2m_{20}m_{22} & m_{21}^2 & 2m_{21}m_{22} & m_{22}^2 \end{bmatrix} \begin{bmatrix} a_{00} \\ a_{01} \\ a_{02} \\ a_{11} \\ a_{12} \\ a_{22} \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \tag{7}$$

The estimated M matrix has information about the rotation and translation which can be used to classify the joint. In the rest of this section, we describe how the information from $\mathbf{M}$ matrix is necessary and sufficient to classify a joint.

Consider motion of two points x_0, x_1 (represented in an inertial frame) on a rigid body at time t_0 and at some subsequent times t_1, t_2. The most general form of rigid body motion of a point can be represented using a rotation matrix $R_{t_0}^{t_1}$ and an associated translation vector $T_{t_0}^{t_1}$ from time instant t_0 to t_1. Motion of any point on that rigid body can be represented as

$$x_0^{t_1} = R_{t_0}^{t_1} x_0^{t_0} + T_{t_0}^{t_1} \tag{8}$$

where the superscript on the point denotes the time.

1) Prismatic: For points lying on a prismatic joint such as a drawer, rotation w.r.t inertial frame remains the same (or the rortation between two frames is identity $R_{t_0}^{t_1} = I$), resulting in $x_1^{t_1} - x_0^{t_1} = x_1^{t_0} - x_0^{t_0}$. This is essentially saying that the vector joining two points on a prismatic joint remains the same before and after the motion.

2) Revolute: For further distinction between revolute and general motion, we need information from more than one time step. For points lying on a body undergoing revolute motion such as a door, the points have same translation vector over time. Hence estimating the translation vector from two time steps $T_{t_0}^{t_1} = T_{t_1}^{t_2}$ is a sufficient condition to classify a joint as revolute joint.

3) Plane Constrained Motion: Plane constrained motion is useful for characterizing motion of objects like chair that can be translated on a plane and rotated about the normal to the plane. Let the plane be denoted by an point x_p lying on the plane and $\hat{n}$ being normal to that plane. Consider the case of a rigid body that has point $x_c^{t_0}$ in contact with the ground plane which after undergoing the motion moves to $x_c^{t_1}$. Since $x_c^{t_0}$ and $x_c^{t_1}$ both lie on the ground plane, we have

$$(x_c^{t_0} - x_0)^T \hat{n} = 0 \tag{9}$$

$$(x_c^{t_1} - x_0)^T \hat{n} = 0 \tag{10}$$

$$x_c^{t_1} = R_{t_0}^{t_1} x_c^{t_0} + T_{t_0}^{t_1} \tag{11}$$

By doing algebraic manipulation we get, $(R_{t_0}^{t_1} x_0 + T - x_0)^T \hat{n} = 0$

4) General Rigid Body Motion: For general motion such as a book that can be rotated and translated anywhere in the space both the rotation and translation matrix will be different.

5) Static: If the rotation matrix between two instances is identity and translation is zero, then the rigid body is stationary.

B. Point Particle Articulation Classification

For the point particle classification, we consider revolute, prismatic and static point types. To find the revolute joint involves finding a circle passing through the observations of the point over time. Similarly, for the prismatic joint, we need to find a line passing through the point particle observation. We will elaborate more on this estimation process in Section XI-A.

VII. Temporal Structure

Articulation estimation provides us with configuration parameters of the articulated motion but one still needs to estimate the evolution of motion variables over time e.g: Position of the object along an axis for prismatic joint. Temporal propagation of articulated bodies will require knowledge of dynamics model parameters (mass, friction etc.) apart from the external excitation (motor torque, force) applied to the system. Several approaches have been proposed for estimating these parameters that use the knowledge of some ground truth trajectories to estimate inertial and friction parameters [22] but they assume apriori access to the object. Furthermore, the external excitation can not be predicted as it can vary depending on the intention of agents.

The goal of our approach is to enforce a structure on the evolution of articulated motion without using any prior information specific to the current articulated body. We take our inspiration from neuroscience literature which posits that humans produce trajectories that are smooth in nature [23] to plan movements from one point to another point in environment. This smoothness assumption can be leveraged by using motion models that use only limited number of derivatives. To concertize, lets assume that $q(t)$ is the articulated motion variable (extension of a prismatic joint, angle of door along a hinge). The system model for a finite order motion model in continuous time domain with $\mathbb{X}(t) = [q, q^1, ., ., q^{n-1}]$ (dropping the explicit time dependence of q and using superscript to denote the order of derivative) as the state can be written as

$$\begin{bmatrix} q^1 \\ q^2 \\ . \\ . \\ q^n \end{bmatrix} = \begin{bmatrix} 0 & 1 & . & . & 0 \\ 0 & 0 & . & . & 0 \\ 0 & 0 & . & . & 0 \\ 0 & 0 & . & . & 1 \\ 0 & 0 & . & . & 0 \end{bmatrix} \begin{bmatrix} q \\ q^1 \\ . \\ . \\ q^{n-1} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ . \\ . \\ 1 \end{bmatrix} \eta \tag{12}$$

where q^n denotes n^{th} order derivative of the motion variable and η is the noise. This state propagation model can be converted to discrete time model as

$$\mathbb{X}(t + \delta t) = A\mathbb{X}(t) + B\eta \tag{13}$$

$$A = \begin{bmatrix} 1 & \delta t & \frac{\delta t^2}{2} & . & . \\ 0 & 1 & \delta t & . & . \\ 0 & 0 & 1 & . & . \\ 0 & 0 & 1 & . & . \\ 0 & 0 & . & . & . \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} B = \begin{bmatrix} \frac{\delta t^n}{n!} \\ \frac{\delta t^{n-1}}{(n-1)!} \\ . \\ . \\ \frac{\delta t^2}{2!} \\ \delta t \end{bmatrix} \tag{14}$$

where A is simply matrix exponential $\exp\{A^c \delta t\}$ of the matrix representation A^c in continuous time equation and $B = (\int_0^{\delta T} \exp\{A^c \mu\} \mathrm{d}\mu) B^c$ where B^c is the continuous time representation.

The kind of model considered in Equation 14 is essentially trying to predict the motion variable $q(t + \delta t)$ using the information at time step t. It is a finite order taylor series expansion of the motion variable $q(t + \delta t)$.

$$q(t + \delta t) = q(t) + \frac{q^1}{1!}\delta t + \frac{q^2}{2!}(\delta t)^2 + ... + \sum_{k=n}^{\infty} \frac{q^k}{n!}(\delta t)^k \tag{15}$$

This representation hence assumes the differentiability of the motion variable. The approximation error in using the order n of the variable is of the order $O((\delta t)^n)$. Various convergence studies can be done to choose the right order n for given time duration δt but here we study the physical aspects of the problem.

A. Choosing Order

Ideally one would want to choose the motion variable order as high as possible to reduce the approximation error as represented in Equation 15 especially for long-term behaviour prediction which is necessary for motion planning or when sensors go blind. But the problem with higher order motion models are due to over-fitting given the need to estimate more parameters from few initial samples. It increases the filtering problem complexity (as described in the later section) significantly as Kalman filtering involves matrix multiplication due to which the computational complexity is atleast $O(n^2)$ where n is the length of state. Furthermore, the error in estimating higher-order derivatives of a noisy signal increases exponentially w.r.t derivative order.

However there are a number of reasons of why we might get away with choosing smaller order of temporal variables. First, in classical mechanics, we only consider second order derivatives of position variables. Also as pointed out earlier, humans minimize jerk [23] in their motion.

VIII. Articulation Model Estimation

We now consider the task of estimating the type of articulated model $M \in \{M_j\}_{j=1}^r$ out of r different models. This does not automatically follow the configuration and motion variables estimation. For example: Consider the case of a point particle moving in $2D$ space, one can fit a line, circle or assume it to be static. One can hypothesize using goodness-of-fit measures to estimate the appropriate model along with some heuristics. However, there are various limitations in comparing goodness-of-fit measures related to number of free parameters in different models, noise in the data, overfitting and number of data samples required [24]. Instead of picking a model at initial time-step, we use a filtering based multiple model approach to correctly pick the model for a given object.

We assume that our target object/particle obeys one of the r ($r \in Z^+, r > 0$) different motion models. In current formulation, we assume a uniform prior $\mu_j(0) = P(M_j), \sum_{j=1}^r \mu_j(0) = 1$ over different motion models for each individual object. This prior can be modified appropriately by object detection such as doors are more likely to have revolute joints etc.. Motion model probability is updated as more and more observations are received [25] as

$$\mu_j(k) \equiv P(M_j|\mathbf{Z}_{0:k}) = \frac{P(z_k|\mathbf{Z}_{0:k-1}, M_j)P(M_j|\mathbf{Z}_{0:k-1})}{P(z_k|\mathbf{Z}_{0:k-1})}$$

$$\mu_j(k) = \frac{P(z_k|\mathbf{Z}_{0:k-1}, M_j)\mu_j(k-1)}{\sum_{j=1}^p P(z_k|\mathbf{Z}_{0:k-1}, M_j)\mu_j(k-1)} \tag{16}$$

The probability of the current observation z_k at time step k, conditioned over a specific articulated motion model and all the previous observation can be represented by various method. In the current work, we filter the states using Extended Kalman Filter, in which this probability is the probability of observation residual w.r.t a normal distribution distributed with zero mean and innovation covariance [25]. To eventually we pick a model when the probability of a particular model becomes greater than a specified threshold.

IX. SLAM for Dynamic World

Figure 2 shows the graphical model of the most general SLAM problem, where x_k, u_k, z_k, m_k, v_k represents the robot state, input to the robot, observation by robot, state of the world and action of various agents in the environment.

Basic SLAM algorithms *assume the map $m_{k-1} \equiv m_k \equiv m$ to be static* and model the combination of robot state and map x_k, m as the state of the estimation problem. The estimation problem only requires motion model $P(x_k|x_{k-1}, u_k)$ and observation model $P(z_k|x_k, m)$. The observation model assumes the observations to be conditionally independent given the the map and the current vehicle state. The goal of the estimation process is to produce unbiased and consistent estimates (expectation of mean squared error should match filter-calculated covariance) [25].

```
Data: $\{M_j\}_{j=1}^r$, $z_k$, $\tau$
Result: $\hat{M} \in \{M_j\}_{j=1}^r$, $C$, $P(q(t+\delta t)|q(t))$
initialization: $C_j = \{\}$, $M = \{\}$ ;
while $M = \{\}$ do
    forall the $M \in \{M_j\}_{j=1}^r$, do
        if $C_j$ is $\{\}$ then
            Estimate $C_j$ ;
            Estimate Temporal Structure ;
        else
            Propagate state using EKF ;
            Estimate $P(z_k|\mathbf{Z}_{0:k-1}, M_j)$ ;
        end
    end
    forall the $M \in \{M_j\}_{j=1}^r$, do
        Normalize to obtain $\mu_j(k)$ ;
        if $\mu_j(k) > \tau$ then
            $\hat{M} = M_j$
        end
    end
end
```

Algorithm 1: Estimating the correct motion model and associated configuration parameters and motion variables

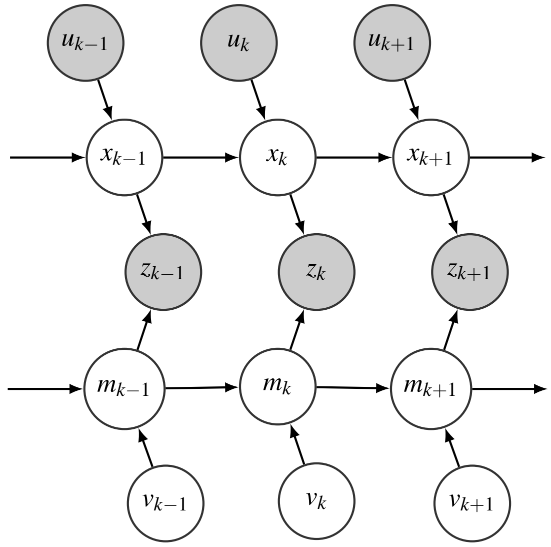


Fig. 2: Graphical Model of the general SLAM problem. The known nodes are darker than the unknown nodes.

For the current SLAM problem, the state consists of time-varying map, (unknown input to the world by various agents) and the robot state. Hence the full estimation problem can be posed as

$$P(x_k, m_k|\mathbf{Z}_{0:k}, \mathbf{U}_{0:k}, \mathbf{V}_{0:k}, x_0, m_0) \tag{17}$$

Following the notation in the review paper on SLAM by Durrant-Whyte and Bailey [26], $\mathbf{Z}_{0:k}$, $\mathbf{U}_{0:k}$ and $\mathbf{V}_{0:k}$ represent the set of observations, robot control inputs and map control inputs from the start time to time step k. It is assumed that the map is markovian in nature which implies that the start state of the map m_0 has all the information needed to make future prediction if actions of various agents in the world $v_{k-1},...,v_{k+1}$ and its impact on the map is known.

A. Time update

The time update models the evolution of state according to the motion model. To write equation concisely, let $A = \{\mathbf{Z}_{0:k-1}, \mathbf{U}_{0:k}, \mathbf{V}_{0:k}, x_0, m_0\}$

$$
\begin{aligned}
&P(x_k, m_k|A) = \\
&\int\int P(x_k, x_{k-1}, m_k, m_{k-1}|A) dx_{k-1} dm_{k-1} \\
&\int\int P(x_k|x_{k-1}, m_k, m_{k-1}, A) P(x_{k-1}, m_k, m_{k-1}|A) dx_{k-1} dm_{k-1} \\
&\int\int P(x_k|x_{k-1}, u_k) P(x_{k-1}, m_k, m_{k-1}|A) dx_{k-1} dm_{k-1} \\
&\int\int P(x_k|x_{k-1}, u_k) P(m_k|x_{k-1}, m_{k-1}, A) P(x_{k-1}, m_{k-1}|A) dx_{k-1} dm_{k-1} \\
&\int\int P(x_k|x_{k-1}, u_k) P(m_k|m_{k-1}, v_{k-1}) P(x_{k-1}, m_{k-1}|A) dx_{k-1} dm_{k-1}
\end{aligned}
\tag{18}
$$

The independence relationship in derivation of time update in Equation 18 are due to the Bayesian networks in Figure 2 in which each node is independent of its non-descendants given the parents of that node. Given the structure of time update, we need two motion models, one for robot: $P(x_k|x_{k-1}, u_k)$ and another one for the world $P(m_k|m_{k-1}, v_{k-1})$. It can be clearly observed that $P(m_k|m_{k-1}, v_{k-1})$ for a static map is dirac delta function and integrates out in Equation 18.

B. Measurement Update

Measurement update uses the bayes formula to update the state of the estimation problem given a new observation z_k at time step k. To write the equations concisely, let $B = \{\mathbf{Z}_{0:k}, \mathbf{U}_{0:k}, \mathbf{V}_{0:k}, x_0, m_0\}$

$$
\begin{aligned}
P(x_k, m_k|B) &= \frac{P(z_k|x_k, m_k, A) P(x_k, m_k|A)}{P(z_k|A)} \\
&= \frac{P(z_k|x_k, m_k) P(x_k, m_k|A)}{P(z_k|A)}
\end{aligned}
\tag{19}
$$

Equation 19 together with equation 18 defines the complete recursive form of the SLAM algorithm for a dynamic environment. Robot motion model and observation model $P(z_k|x_k, m_k)$ are well described in previous literature and hence we will exclude that from current discussion. The focus of current work is the representation of map motion model to extend the standard SLAM algorithm with its static world assumption to dynamic world.

X. ARTICULATED EKF SLAM

A. Robot Motion Model

We consider a robot with state $x_k = (x, y, \theta)^T$ at time k moving with constant linear velocity v_k and angular velocity ω_k. The state of the robot at next time step can be represented as

$$
x_{k+1} = \begin{pmatrix} x - \frac{v_k}{\omega_k}\sin\theta + \frac{v_k}{\omega_k}\sin(\theta + \omega_k \delta t) \\ y + \frac{v_k}{\omega_k}\cos\theta - \frac{v_k}{\omega_k}\cos(\theta + \omega_k \delta t) \\ \theta + \omega_k \delta t \end{pmatrix} + \mathcal{N}(0, R_k) \tag{20}
$$

, where δt is the time step and R_k is the error covariance of noise which is distributed with a zero mean Gaussian. Error covariance can be derived by propagating the noise in input to the state space[27].

If the angular velocity is close to zero, the robot model as represented in Equation 21 will be ill-conditioned. The model with zero angular velocity is given by

$$
x_{k+1} = \begin{pmatrix} x + v_k \delta t \cos(\theta) \\ y + v_k \delta t \sin(\theta) \\ \theta \end{pmatrix} + \mathcal{N}(0, R_k) \tag{21}
$$

Following the approximations proposed by Thrun et. al [27], the angular and linear velocities are generated by by a motion control unit $\hat{u}_k = (\hat{v}_k, \hat{\omega}_k)^T$ with zero mean additive Gaussian noise.

$$
\begin{pmatrix} v_k \\ \omega_k \end{pmatrix} = \begin{pmatrix} \hat{v}_k \\ \hat{\omega}_k \end{pmatrix} + \mathcal{N}(0, M_k) \tag{22}
$$

$$
M_k = \begin{pmatrix} \alpha_1 \hat{v}_k^2 + \alpha_2 \hat{\omega}_k^2 & 0 \\ 0 & \alpha_3 \hat{v}_k^2 + \alpha_4 \hat{\omega}_k^2 \end{pmatrix} \tag{23}
$$

where $\alpha_1, \alpha_2, \alpha_3, \alpha_4$ are the noise coefficients.

B. Observation Model

Robot measures range and bearing angle for j^{th} landmark located at position $m_j = (m_{j,x}, m_{j,y})^T$ within robot's sensor field of view. Each observation can be written as

$$
z_k^i = \begin{pmatrix} \sqrt{(m_{j,x} - x)^2 + (m_{j,y} - y)^2} \\ \arctan(m_{j,y} - y, m_{j,x} - x) - \theta \end{pmatrix} + \mathcal{N}(0, Q_k) \tag{24}
$$

, where z_k^i is the i^{th} observation at time step k which is also affected by zero mean Gaussian noise with covariance matrix Q_k.

C. Jacobian Computation

Extended Kalman Filtering (EKF) requires linearization to ensure that the state propagation and observation assimilation maintains Gaussianity of the state distribution. In order to propagate state, we need to estimate the jacobian of the state propagation model w.r.t to state at time step k. The state jacobian can be represented as

$$
J_{x_k}^{x_{k+1}} = \begin{pmatrix} 1 & 0 & -\frac{v_k}{\omega_k}\cos\theta + \frac{v_k}{\omega_k}\cos(\theta + \omega_k \delta t) \\ 0 & 1 & -\frac{v_k}{\omega_k}\sin\theta + \frac{v_k}{\omega_k}\sin(\theta + \omega_k \delta t) \\ 0 & 0 & 1 \end{pmatrix} \tag{25}
$$

Furthermore the error in the input control space needs to be projected to the state space for which one needs to compute jacobian of state propagation model w.r.t input u_k.

$$
J_{u_k}^{x_{k+1}} = \begin{pmatrix} \frac{-\sin\theta + \sin(\theta + \omega_k \delta t)}{\omega_k} & \frac{v_k(\sin\theta - \sin(\theta + \omega_k \delta t))}{\omega_k^2} + \frac{v_k \cos(\theta + \omega_k \delta t)}{\omega_k} \\ \frac{\cos\theta - \cos(\theta + \omega_k \delta t)}{\omega_k} & \frac{-v_k(\cos\theta - \cos(\theta + \omega_k \delta t))}{\omega_k^2} + \frac{v_k \sin(\theta + \omega_k \delta t)}{\omega_k} \\ 0 & \delta t \end{pmatrix} \tag{26}
$$

To assimilate each observation z_i^k, we need to compute jacobian of the observation model w.r.t to the overall SLAM state which consists of robot state as well as motion parameters state associated with each landmark. However for computing the jacobian matrix for i^{th} observation at time step k of landmark j, the only relevant entries in the jacobian matrix are derivative of observation w.r.t robot states and motion parameters state associated with landmark j. Jacobian of observation w.r.t robot state is

$$J^{z^i_k}_{x_k} = \begin{pmatrix} \frac{x-m_{j,x}}{\sqrt{q}} & \frac{y-m_{j,y}}{\sqrt{q}} & 0 \\ \frac{m_{j,y}-y}{q} & \frac{x-m_{j,x}}{q} & -1 \end{pmatrix} \tag{27}$$

and the jacobian w.r.t motion parameters state is

$$J^{z^i_k}_{m_j} = \begin{pmatrix} \frac{m_{j,x}-x}{\sqrt{q}} & \frac{m_{j,y}-y}{\sqrt{q}} \\ \frac{y-m_{j,y}}{q} & \frac{m_{j,x}-x}{q} \end{pmatrix} J^{m_j}_{m(t)} \tag{28}$$

where $J^{m_j}_{m(t)}$ is the jacobian of landmark observation w.r.t motion parameters state $m(t)$.

```
Data: $\mu_{t-1}$, $\Sigma_{t-1}$, $u_t$, $\{M_j\}_{j=1}^{r}$, $z_k$, $\tau$
Result: $\mu_t$, $\Sigma_t$
Propagate Robot State and Covariance;
Propagate Landmarks State and Covariance;
forall the $z_k^i \in z_k$ do
    if $\hat{M} \neq \{\}$ then
        Estimate Motion Model($\{M_j\}_{j=1}^{r}$, $z_k$, $\tau$);
    else
        Assimilate Observation;
    end
end
```

Algorithm 2: Articulated EKF SLAM

XI. Results

A. Configuration Estimation

We tested our configuration estimation for a variety of joint models. To elucidate the effectiveness of the separating modeling of motion parameters from configuration parameters, we consider the configuration estimation of revolute motion. Consider a point moving along a circle centered at $X_c = [x_c, y_c]^T$ with a radius r. According to our definition configuration here refers to $C = [x_x, y_c, r]^T$ while the motion parameter is θ which represents the time-varying angle made by the moving point. For joint estimation, one first needs to assume the order of motion prior to configuration estimation. For current case, we assume a constant-velocity motion model which can be written as

$$X(t) = X_c + r[\cos(\theta_0 + \Delta T\omega), \sin(\theta_0 + \Delta T\omega)]^T \tag{29}$$

where $X(t)$ is the observed motion, θ_0 is the initial angle and ω is the constant angular velocity of the point. As can be observed the estimation problem resulting from Eq. 29 is non-linear in θ_0 and ω. For the separate representation the estimation problem reduces to estimation of a circle from points lying on a circle which is linear. Figure 3 shows the estimation results. To estimate the resulting errors, we did a monte-carlo simulation and averaged errors across all the trials. Table I shows results from 500 monte-carlo runs of the algorithm for the same problem. The errors in estimation is L_2 norm of difference between estimated and true center and radius. It can be observed that separate estimation has considerably less errors compared to joint estimation problem.

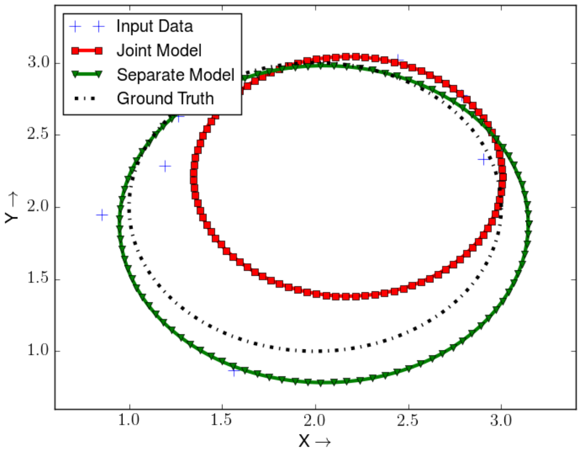


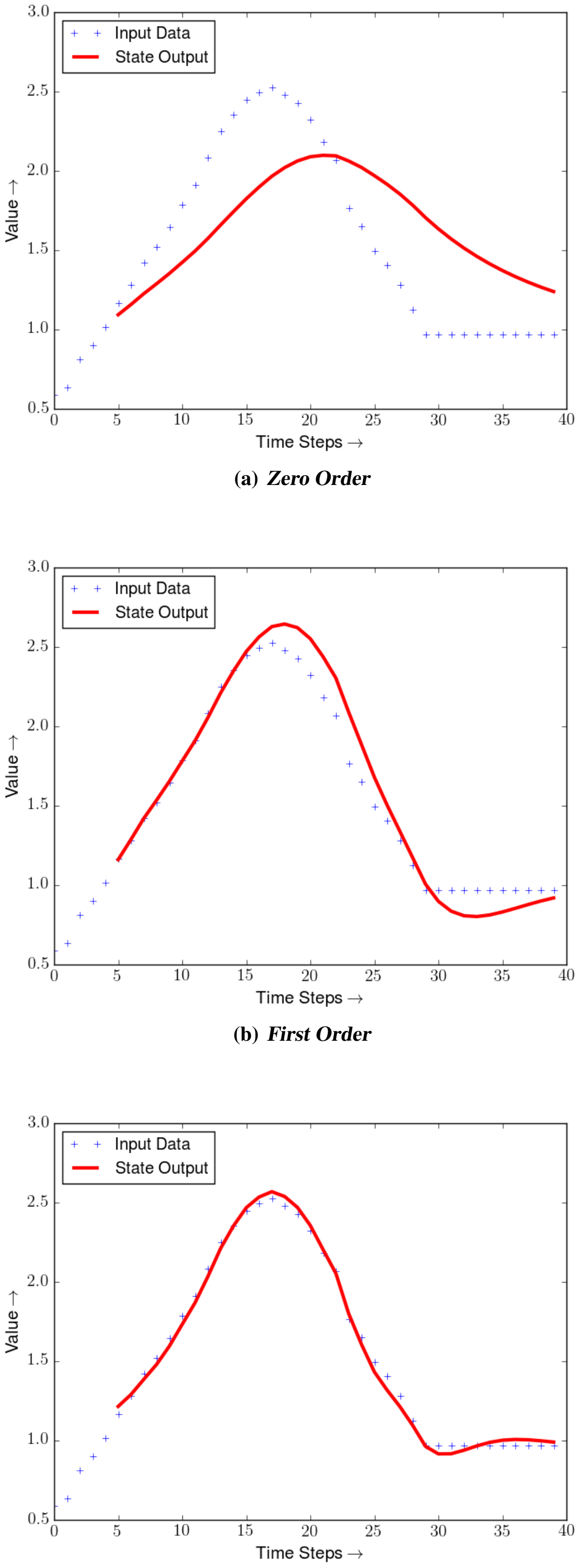
Fig. 3: Estimation of configuration parameters for a 2D landmark in revolute motion centered at point $(2,2)$ with radius 1. Gaussian noise of 0.01 variance in both X and Y directions. Joint estimation yield a revolute motion centered at $(2.18, 2, 21)$ with a radius of 0.83 while separate configuration estimation yields in a revolute joint centered at $(2.05, 1.88)$ with a radius of 1.10

Estimation	Center Error	Radius Error
Joint	0.71	0.09
Separate	0.60	0.04

TABLE I: Error metrics for center and radius error

B. Temporal Order

Once the configuration parameters have been estimated, various orders of motion models can be estimated from motion data. In order to perform this, we assume that there is a function $G : (X(t), C) \mapsto m(t)$ which can map observation and configuration data to motion parameters. This allows us to obtain motion parameters over time to which various order of motion can be fitted. We took the raw angular trajectory of a pendulum and fitted zeroth, first and second order motion models. For a zeroth, first and second order motion model the state is θ, $[\theta, \dot{\theta}]$, $[\theta, \dot{\theta}, \ddot{\theta}]$ where $\theta, \dot{\theta}, \ddot{\theta}$ are zeroth, first and second order derivative of the motion parameter. The motion parameter can be propagated to next time frame using the framework in Section VII. For the observation model, we assumed the observation of motion parameter itself which is equivalent to observation of the body $X(t)$ once the configuration is estimated using the function G. Figure 4 shows the motion parameter for various different orders. It can be observed that higher order motion model clearly follow the trajectory much better than lower order motion models.



(a) ***Zero Order***

(b) ***First Order***

(c) ***Second Order***

Fig. 4: Comparison of EKF filtering based state estimation for various orders of a motion parameter. For displaying purposes, we only show the zeroth order derivative state from all the different motion models.

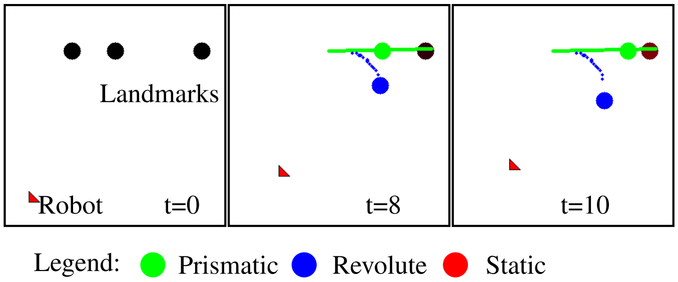


Fig. 5: Frames at different time intervals of our simulation. Color of a landmark at a particular frame is the weighted sum of colors assigned to each motion model. The weights used are the probability of the landmark following that particular motion model and estimated by our algorithm. We also show the predicted trajectory of a landmark according to the estimated motion model.

C. Articulation Estimation

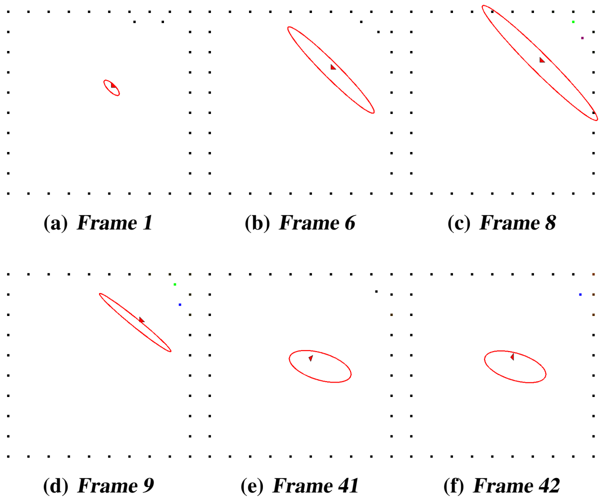
To test articulation estimation, we simulated an environment with one static, prismatic and revolute points each. We use a minimum of 7 samples to estimate configuration and initialize motion parameters. Figure 5 shows the results for the articulation estimation. In long term, all the articulations models are estimated correctly. However as can be observed Static articulation takes the longest time to be correctly estimated. This is because of difficulty in separating static landmark from a revolute landmark with 0 radius and 0 velocity and a prismatic landmark with 0 velocity.

D. Dynamic World SLAM

We simulated a map with point features that are either static, prismatic or revolute. A robot with limited field of view simulated reading from a laser scanner which were then used to simultaneously localize the robot as well map the environment.

1) Qualitative Analysis: We used a total of 42 landmarks in the scene with 1 revolute, 1 prismatic and 40 static landmark. Both the revolute and prismatic landmarks were correctly identified while our algorithm could not identify a total of 2 static landmark (from have been observed more than 10 times by the robot) with a selection threshold of $\tau = 0.75$. With a relaxed selection threshold of $\tau = 0.5$, our algorithm correctly identified all the static landmarks with more than 10 samples. observations. Figure 6 shows the summary of results from the articulated EKF algorithm. Our algorithm needs a minimum of 7 samples and hence till frame 7, the covariance of the robot state keeps increasing. On 8^{th} frame, our estimation algorithm correctly identifies two landmarks and as a result the covariance of the robot state decreases.

2) Quantitative Analysis: We compared the proposed Articulated EKF SLAM algorithm against the standard EKF SLAM algorithm. In contrast to our algorithm where the SLAM state includes the motion parameters, standard EKF SLAM algorithm includes the landmark position in the state. As a result, we are only comparing the resulting localization estimates of the robot. Notably, only two landmarks are non-static landmarks which violate the assumption of standard SLAM algorithms. Table II summarizes the average localiza-



(a) *Frame 1* (b) *Frame 6* (c) *Frame 8*

(d) *Frame 9* (e) *Frame 41* (f) *Frame 42*

Fig. 6: Demonstration of Articulated EKF algorithm at various time steps. At each time step, we plot the robot true state with a triangle and the estimation of robot's mean and covariance SLAM states is shown by an ellipse.

Algorithm	Avg. Translation Error	Avg. Rotation Error
Articulated EKF SLAM	1.592	0.076
EKF SLAM	3.652	0.110

TABLE II: Comparison of Localization Error for two different SLAM algorithms

tion error metrics in both position and orientation increments [28]. As is evident from results, our algorithm clearly improves on both the error metrics significantly even with just two landmarks in motion.

References

[1] Shuang Wu, Hongjing Lu, and Alan L Yuille. Model selection and velocity estimation using novel priors for motion patterns. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems 21*, pages 1793–1800. Curran Associates, Inc., 2009.

[2] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jagannath Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 580–587. IEEE, 2014.

[3] Yi Yang and Deva Ramanan. Articulated pose estimation with flexible mixtures-of-parts. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 1385–1392. IEEE, 2011.

[4] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[5] Changhyun Choi and Henrik I Christensen. 3d textureless object detection and tracking: An edge-based approach. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 3877–3884. IEEE, 2012.

[6] Shimon Ullman. *The interpretation of visual motion.* MIT Press, 1979.

[7] Carlo Tomasi and Takeo Kanade. Shape and motion from image streams under orthography: a factorization method. *International Journal of Computer Vision*, 9(2):137–154, 1992.

[8] João Paulo Costeira and Takeo Kanade. A multibody factorization method for independently moving objects. *International Journal of Computer Vision*, 29(3):159–179, 1998.

[9] Jingyu Yan and M. Pollefeys. Automatic kinematic chain building from feature trajectories of articulated objects. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 1, pages 712–719, June 2006.

[10] Sudeep Pillai, Matthew Walter, and Seth Teller. Learning articulated motions from visual demonstration. In *Proceedings of Robotics: Science and Systems*, Berkeley, USA, July 2014.

[11] Dov Katz, Moslem Kazemi, J. Andrew (Drew) Bagnell, and Anthony (Tony) Stentz. Interactive segmentation, tracking, and kinematic modeling of unknown 3d articulated objects. In *Proceedings of IEEE International Conference on Robotics and Automation*, May 2013.

[12] Mark Fiala. Comparing artag and artoolkit plus fiducial marker systems. In *Haptic Audio Visual Environments and their Applications, 2005. IEEE International Workshop on*, pages 6–pp. IEEE, 2005.

[13] Steven Gray, Subhashini Chitta, Vipin Kumar, and Maxim Likhachev. A single planner for a composite task of approaching, opening and navigating through non-spring and spring-loaded doors. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 3839–3846. IEEE, 2013.

[14] Jürgen Sturm, Cyrill Stachniss, and Wolfram Burgard. A probabilistic framework for learning kinematic models of articulated objects. *Journal of Artificial Intelligence Research*, pages 477–526, 2011.

[15] Jürgen Sturm. Learning kinematic models of articulated objects. In *Approaches to Probabilistic Model Learning for Mobile Manipulation Robots*, pages 65–111. Springer, 2013.

[16] Karol Hausman, Scott Niekum, Sarah Osentoski, and G Sukhatme. Active articulation model estimation through interactive perception. In *submitted to) IEEE International Conference on Robotics and Automation (ICRA)*, 2015.

[17] Dov Katz and Oliver Brock. Extracting planar kinematic models using interactive perception. In *Unifying Perspectives in Computational and Robot Vision*, pages 11–23. Springer, 2008.

[18] Roberto Martin Martin and Oliver Brock. Online interactive perception of articulated objects with multi-level recursive estimation based on task-specific priors. In *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, pages 2494–2501. IEEE, 2014.

[19] Tim Bailey and Hugh Durrant-Whyte. Simultaneous localization and mapping (slam): Part ii. *IEEE Robotics & Automation Magazine*, 13(3):108–117, 2006.

[20] Tim Bailey. *Mobile robot localisation and mapping in extensive outdoor environments*. PhD thesis, Citeseer, 2002.

[21] Chieh-Chih Wang, Charles Thorpe, and Sebastian 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 *Robotics and Automation, 2003. Proceedings. ICRA'03. IEEE International Conference on*, volume 1, pages 842–849. IEEE, 2003.

[22] Felix Endres, Jeff Trinkle, and Wolfram Burgard. Learning the dynamics of doors for robotic manipulation. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pages 3543–3549. IEEE, 2013.

[23] Tamar Flash and Neville Hogan. The coordination of arm movements: an experimentally confirmed mathematical model. *The journal of Neuroscience*, 5(7):1688–1703, 1985.

[24] Christian D Schunn and Dieter Wallach. Evaluating goodness-of-fit in comparison of models to data. *Psychologie der Kognition: Reden and vorträge anlässlich der emeritierung von Werner Tack*, pages 115–154, 2005.

[25] Bar-Shalom Yaakov, XR Li, and Kirubarajan Thiagalingam. Estimation with applications to tracking and navigation. *New York: Johh Wiley and Sons*, 245, 2001.

[26] Hugh Durrant-Whyte and Tim Bailey. Simultaneous localization and mapping: part i. *Robotics & Automation Magazine, IEEE*, 13(2):99–110, 2006.

[27] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic robotics*. MIT press, 2005.

[28] Rainer Kümmerle, Bastian Steder, Christian Dornhege, Michael Ruhnke, Giorgio Grisetti, Cyrill Stachniss, and Alexander Kleiner.

On measuring the accuracy of slam algorithms. *Autonomous Robots*, 27(4):387–407, 2009.