Global Localization of Objects via Touch

Anna Petrovskaya and Oussama Khatib

***Abstract*—Humans are capable of manipulating objects based solely on the sense of touch. For robots to achieve the same feat in unstructured environments, global localization of objects via touch is required. Bayesian approaches provide means for coping with uncertainties of the real world, but estimation of the Bayesian posterior for the full 6DOF global localization problem is computationally prohibitive. We propose an efficient Bayesian approach termed Scaling Series. It is capable of solving the full problem reliably in real time. This is a Monte Carlo approach, that performs a series of successive refinements coupled with annealing. We also propose an analytical measurement model, which can be computed efficiently at run time for any object represented as a polygonal mesh. Extensive empirical evaluation shows that Scaling Series drastically outperforms prior approaches. We demonstrate general applicability of the approach on five common solid objects, which are rigidly fixed during the experiments. We also consider 6DOF localization and tracking of free standing objects that can move during tactile exploration.**

***Index Terms*—force and tactile sensing, localization, Bayesian state estimation**

I. Introduction

In ORDER to carry out manipulation tasks in real world environments, robots need to perceive objects around them based on sensory information. Although for robots the use of vision has been studied in more depth [1], humans rely heavily on the sense of touch for manipulation tasks [2]. In fact humans are capable of manipulating objects based solely on the sense of touch. Working towards this ability in robots, we consider global localization of solid objects via touch (Fig. 1). Gaining this ability would allow robots to operate in environments where vision is not available, such as smoke filled rooms or muddy water, or it could be used in combination with vision to improve overall perception.

Early tactile perception algorithms date back to the 1980s (e.g. [3]–[5]) as we discuss in the next section. Recent work has focused on tactile perception in uncertain environments. However, in uncertain conditions object localization requires the estimation of a probability distribution over the space of all 6DOF[1] poses of the object. When initial uncertainty is high, this estimation is very expensive computationally. For this reason, most approaches limit the DOFs and/or initial uncertainty [6]–[8].

To overcome the computational challenge, we propose a principled approach — termed Scaling Series (SS) — that solves the full global 6DOF localization problem efficiently ($\sim$ 1 second) and reliably ($\geq$ 99%). The approach is a Bayesian Monte Carlo technique coupled with annealing. It performs multiple iterations over the data, gradually scaling the precision from low to high. For each iteration, the number of particles is selected automatically based on the complexity of the annealed posterior. [2]

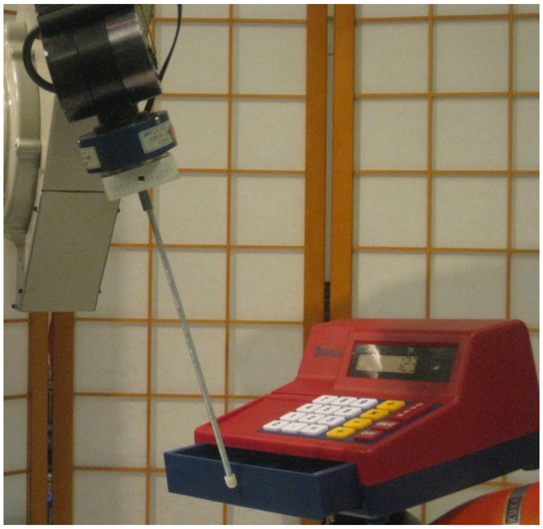


Fig. 1. In our experiments robots manipulate objects based solely on the sense of touch. Global initial uncertainty is assumed in 6DOF. The photo shows the robot interacting with one of the five objects used in our experiments: the cash register.

We show that Scaling Series works in both fully-constrained uni-modal scenarios and under-constrained multi-modal scenarios. The latter arise at early stages of tactile exploration, when insufficient data have been collected to fully constrain the problem. We also consider free-standing objects, which can move during tactile exploration. To our knowledge, full 6DOF Bayesian estimation for this case has not been addressed in prior art.

In addition, we present an analytical measurement model for tactile perception that can be used for any object represented as a polygonal mesh. Unlike sampling based models, this model can be computed quickly at run time and does not require training ahead of time. Due to its differentiability, the presented model allows for efficient estimation.

Our approach is easily applicable to any object represented as a polygonal mesh. We demonstrate its portability on five common rigid objects (Fig. 2). High initial uncertainty is assumed in the experiments: 400mm in position with unre-

A. Petrovskaya and O. Khatib are with the Computer Science Department, Stanford University, Stanford, CA 94305, USA e-mail: anya@cs.stanford.edu, ok@cs.stanford.edu.

Manuscript received August 11, 2010; revised January 23, 2011; accepted March 28, 2011. The official version of this paper is available online at http://ieeexplore.ieee.org.

[1] 6DOF stands for six degrees of freedom.

[2] Earlier versions of this paper appeared at ICRA 2006 [9] and at an RSS workshop in 2007 [10]. In addition to the material presented in earlier versions, we provide a sound theoretical foundation for the approach including proofs of convergence and considerations for parameter selection. We also provide an in-depth evaluation of the algorithm features and significantly expand comparisons to prior art. Moreover, we include results for three new complex objects and consider moving objects for the first time.

stricted orientation. The presented approach produces highly accurate results ($\sim$ 1mm) quickly and reliably, enabling the robots to safely manipulate the objects. We also provide extensive empirical evaluation of Scaling Series properties and provide comparisons to other methods, including particle filters, importance sampling, and APF.

The paper is organized as follows. The next section discusses related work. Sect. III provides the necessary mathematical background. Sect. IV presents the algorithm together with its discussion and analysis. Experimental results are presented in Sect. V. We conclude in Sect. VI. Mathematical derivations are provided in the Appendix.

II. Related Work

Touch based perception has not been studied in as much depth as vision because standardized touch sensors are not as easily available. In many situations tactile sensors have to be hand crafted specifically for the robot and the task. This complicates comparisons between methods and slows progress in tactile perception. However, recently there has been a surge of interest in the field due to the necessity of touch based perception in service applications [8], [11]–[13].

A. Single Hypothesis Methods

Early methods for tactile object localization generally ignore the sensing process uncertainties and focus on finding a single hypothesis that best fits the measurements. For example, in 1983 Gaston *et al.* used interpretation trees to efficiently find the best match for 3DOF object localization [14]. Grimson *et al.* extended the approach to 6DOF [3]. Faugeras *et al.* used least squares to perform geometrical matching between primitive surfaces [4]. In 1986 Shekhar *et al.* solved systems of weighted linear equations to localize an object held in a robotic hand [5].

Single hypothesis methods are also widely used to solve the *workpiece localization* problem in manufacturing applications for dimensional inspection [15], machining [16], and robotic assembly [17]. In these applications the measurements are taken by a *coordinate measurement machine* (CMM) [18] or by on-machine sensors [19]. Workpiece localization makes a number of restrictive assumptions, which make it inapplicable to autonomous robot operation in unstructured environments. One important restriction is that there is a known correspondence between each measured data point and a point or patch on the object surface (called *home point* or *home surface* respectively) [20]. In semi-automated settings the correspondence assumption is satisfied by having a human direct the robot to specific locations on the object. In fully-automated settings the object is placed on the measurement table with low uncertainty to make sure each data point lands near the corresponding home point.

Further restrictions include assumptions that the data are sufficient to fully constrain the object, the object does not move, and there are no unmodeled effects (e.g. vibration, deformation, or temperature variation). All of these parameters are carefully controlled for in the structured manufacturing environments.

The workpiece localization problem is usually solved in least squares form using iterative optimization methods, including Hong-Tan method [21], Variational method [22], and Menq method [23]. Since these methods are prone to getting trapped in local minima, low initial uncertainty is usually assumed to make sure the optimization algorithm is initialized near the solution. Some attempts have been made to solve the global localization problem by re-running the optimization algorithm multiple times from pre-specified and random initial points [24]. Recent work has focused on careful selection of the home points to improve localization results [25]–[27] and on improving localization efficiently with complex home surfaces [28], [29].

B. Bayesian Methods in Tactile Perception

In the last decade there has been increased interest in Bayesian state estimation for the tactile object localization problem [6]–[8], [30]. These methods estimate the probability distribution over all possible states (the posterior), which captures the uncertainty resulting from noisy sensors, inaccurate object models, and other effects present during the sensing process. Thus estimation of the posterior enables planning algorithms that are resilient to the uncertainties of the real world. Unlike workpiece localization, these methods do not assume known correspondence. In contrast to single hypothesis methods, posterior estimation methods can handle the under constrained scenario, in which the data are insufficient to fully localize the object. These methods can also work with moving objects and answer important questions, such as:"have we localized the object completely?" and "where is the best place to sense next?".

The main challenge faced by posterior estimation approaches is computational complexity, which goes up exponentially with the number of DOFs and the size of the initial uncertainty region. For this reason all of the approaches in this category (except an earlier version of this paper [9]) restrict the number of DOFs and/or initial uncertainty.

The earliest known work in this category was in 2001 by Gadeyne *et al.*, who considered localization of a rectangular box based on measurements taken by a force controlled robot [6]. The localization was performed in 3DOF with initial uncertainty of 300mm in position and 360° in orientation. They used a sampled measurement model that was stored in a look-up table.

In 2005 Chatpar *et al.* used particle filters for contact based object localization during peg-in-hole assembly tasks [7]. They considered 20mm initial uncertainty in 3DOF and utilized a measurement model based on sampling the object in advance. Chatpar *et al.* also considered active localization, where the most optimal next sensing action is chosen based on information from prior steps.

An earlier version of this paper was published in 2006 [9]. We considered 6DOF localization with large uncertainty: 400mm in position and 360° in orientation. We also introduced an analytical measurement model and proposed the Scaling Series method.

In 2010 Corcoran *et al.* used the annealed particle filter (APF) to estimate 4DOF pose and radius of cylindrical objects

[30]. Initial uncertainty of up to 250mm in position with unrestricted orientation was considered. They also extended the analytical measurement model we proposed in 2006 to include some negative information and to integrate over object surface. Later in 2010, Platt *et al.* introduced sample based models suitable for localization of deformable objects [31].

Most recently Hsiao *et al.* used grids to estimate the posterior in 3DOF with low-to-medium initial uncertainty (up to 50mm Gaussian) [8]. The contribution of their approach was in optimizing data collection strategies and considering free standing objects that could potentially move during data collection. The measurement model used in their work is similar to the one we proposed in 2006, except that it also takes negative information into account.

We should also mention the rich literature on object shape reconstruction using tactile sensors [32]–[35]. Although this work does not address localization of known objects, some authors explicitly consider sensor uncertainties using Bayesian methods [36], [37].

C. Bayesian Methods in Other Applications

Bayesian methods have been used in a variety of robotic applications with great success. For example a recent book on practical applications includes analysis of planetary ring structure, shape estimation, and target tracking to name a few [38]. A recent textbook [39] provides an in-depth study of indoor robot localization and mapping, which bear some resemblance to the problems considered in this paper. However, the global localization problems considered in the textbook are relatively low dimensional: 3DOF. The only high dimensional problem considered in the textbook is simultaneous localization and mapping (SLAM), where global uncertainty does not need to be resolved. Moreover many SLAM methods effectively reduce dimensionality by utilizing problem structure [40]. These techniques do not apply to the 6DOF object state estimation problem, where this structure is not present.

There has been a lot of work on 6DOF object localization in the vision community. See [41] for a recent survey by Lepetit and Fua. The most popular methods have been least-squares minimization [42], [43], RANSAC [44], Kalman filter variants [45], [46], and particle filters [47]. These approaches tend to rely on manual initialization and assume small initial uncertainty. As Lepetit and Fua point out, methods incapable of dealing with global uncertainty tend to be inherently fragile because they can not recover from tracking failures.

One of the most successful variants of particle filters, the annealed particle filter (APF), has been introduced by Deutscher *et al.* in the context of articulated body tracking using vision [48], [49]. As we already mentioned above in Sect. II-B, this method has also been applied to the tactile localization problem [30]. Articulated object tracking is a very high dimensional problem (up to 30DOF). However, usually low initial uncertainty is assumed in these applications, due to the use of manual initialization. Also these approaches do not run in real time. APF tends to outperform the standard particle filter in single-mode scenarios. However, it has been shown to be unstable in multi-modal situations by Balan *et al.* [50]. In fact Balan *et al.* argue for the use of standard particle filters instead of APF for this very reason.

III. Mathematical Background

We start out with a quick intuitive summary of the problem: tactile object localization requires estimation of state parameters based on a set of data obtained by touching the object. As we shall see in Sect. III-C this entails fitting the data to the object model using Mahalonobis distance in the 6-D measurement space. In the case of moving objects, the estimation is performed via recursive filtering from one time step to the next.

Instead of producing a single set of parameter values, Bayesian approaches represent the uncertain knowledge by a probability distribution, which records how likely each state is based on sensor measurements. Estimating the entire probability distribution over all the states is important because initially the data are insufficient to disambiguate the object's position. In fact, the shape of the probability distribution (specifically the high likelihood regions, called *modes*) allows us to determine when enough data has been collected in order to manipulate the object safely. The probability distribution is represented numerically by weighted points, called *particles*.[3]

In the remainder of this section we formalize the above intuitive description and introduce the required notation.

A. Bayesian Problem Statement and Definitions

We consider the class of problems where the state X has to be inferred from a set of sensor measurements $\mathcal{D} = \{Y_k\}$. Our goal is to estimate the probability distribution of the state given the measurements, $bel(X) := p(X|\mathcal{D})$, known as the *posterior distribution*, which represents our uncertain belief about the state X.

For the general algorithm, we will assume that the state X is a vector of dimensionality $dimX$ in $\mathbb{R}^{dimX}$. The measurements are modeled as K random variables Y_k, which are drawn independently from conditional probability distributions $p(Y_k|X)$ with domains in $\mathbb{R}^{dimY}$. The conditional probability distributions (CPDs) encode the *measurement model*, which is a probabilistic law that represents the measurement process. The measurement model depends non-linearly on the state X. In many applications, the CPDs are naturally given in the log-linear form via *measurement energy potentials* v_k : $\mathbb{R}^{dimX} \times \mathbb{R}^{dimY} \mapsto \mathbb{R}^+$. Then the CPD for Y_k can be written as

$$p(Y_k|X) = \eta \exp\Big(-v_k(X, Y_k)\Big). \tag{1}$$

In the above equation and throughout the paper η denotes the normalization constant, whose value is such that the expression integrates to 1. We also define the *total measurement energy*

$$v(X) := \sum_k v_k(X, Y_k). \tag{2}$$

Via Bayes rule the posterior $bel(X)$ can be shown to be proportional to $p(\mathcal{D}|X)p(X)$. The first factor is the *data probability*, which can be shown to be proportionate to

[3] See [51] for further information on particle based Bayesian methods.

$\pi(X) := \exp(-v(X))$. The second factor, $\overline{bel}(X) := p(X)$, is called the *prior*, which represents our belief about X before obtaining measurements $\mathcal{D}$. Hence with this notation we can write

$$bel(X) = \eta\pi(X)\overline{bel}(X). \tag{3}$$

1) Stationary systems: In stationary systems with global initial uncertainty the prior $\overline{bel}(X)$ is uniform. Hence, the posterior is proportional to the data probability: $bel(X) = \eta\pi(X)$.

2) Dynamic systems: In dynamic systems the state changes over time. In this case, X_t and $\mathcal{D}_t$ denote the state and the set of sensor measurements for a time step t. The posterior, bel_t, is defined as the probability of the current state given all measurements obtained up until this point:

$$bel_t\left(X_t\right) := p\left(X_t|\mathcal{D}_1, \cdots, \mathcal{D}_t\right). \tag{4}$$

Measurement CPDs for step t are defined analogously to Eqn. 1. Similarly define $\pi_t(X_t) := \exp(-v_t(X_t))$. Also let $\overline{bel}_t(X_t)$ be the prior at time t. For brevity we will drop the argument X_t and write bel_t, π_t, and $\overline{bel}_t$ to denote the values of these functions at time t.

In dynamic systems, the prior is the prediction distribution, $\overline{bel}_t := p\left(X_t|\mathcal{D}_1, \cdots, \mathcal{D}_{t-1}\right)$, which predicts the current state X_t before taking into account the most recent sensor data $\mathcal{D}_t$. Hence the prior is computed as

$$\overline{bel}_t = \int p\left(X_t|X_{t-1}\right) bel_{t-1}\, \mathrm{d}X_{t-1}. \tag{5}$$

Here $p\left(X_t|X_{t-1}\right)$ encodes the dynamics of the system. This probability is called the *motion model*. Combining Eqns. 3 and 5 we obtain the *Bayesian recursion* equation:

$$bel_t = \eta\, \pi_t \int p\left(X_t|X_{t-1}\right) bel_{t-1}\, \mathrm{d}X_{t-1}. \tag{6}$$

B. Problem Statement for Tactile Localization

Bayesian tactile localization is an instance of the general Bayesian problem defined in the previous section. Here the robot needs to determine the pose X of a known object $\mathcal{O}$ based on a set of tactile measurements $\mathcal{D}$. The object is typically represented as a polygonal mesh (Fig. 2). The state $X := (x, y, z, \alpha, \beta, \gamma)$ is the 6DOF pose of the object — including position (x, y, z) and orientation angles (α, β, γ) — in the manipulator coordinate frame. The measurements $\mathcal{D}$ are obtained by touching the object with the robot's end effector. Each measurement $Y_k := (Y_k^{\mathrm{pos}}, Y_k^{\mathrm{nor}})$ consists of the measured cartesian position of the contact point Y_k^{pos} and the measured surface normal Y_k^{nor}.

Note that unlike in the workpiece localization problem, here we do not assume known correspondence between measurements and points on the surface of the object. Hence the resulting problem is more complex than workpiece localization.

C. Measurement Model

To interpret the tactile measurements we use the proximity measurement model, which has been used in stereo vision [52] and is known as "likelihood fields" in mobile robotics [39]. In this model the measurements are considered independent of each other with both position and normal components corrupted by Gaussian noise. For each measurement, the potential depends on the distance between the measurement and the object (hence the name "proximity").

Since the measurements contain both contact coordinates and surface normals, this distance is taken in the 6-D space of coordinates and normals (i.e. in the measurement space). Let $\hat{\mathcal{O}}$ be a representation of the object in this 6-D space. Let $\hat{o} := (\hat{o}^{\mathrm{pos}}, \hat{o}^{\mathrm{nor}})$ be a point on the object surface, and Y be a measurement. Define $D_{\mathcal{M}}(\hat{o}, Y)$ to be the Mahalonobis distance between $\hat{o}$ and Y:

$$D_{\mathcal{M}}(\hat{o}, Y) := \sqrt{\frac{||\hat{o}^{\mathrm{pos}} - Y^{\mathrm{pos}}||^2}{\sigma^2_{\mathrm{pos}}} + \frac{||\hat{o}^{\mathrm{nor}} - Y^{\mathrm{nor}}||^2}{\sigma^2_{\mathrm{nor}}}}, \tag{7}$$

where σ^2_{pos} and σ^2_{nor} are Gaussian noise variances of position and normal measurement components respectively. Then the distance between a measurement Y and the object is $D_{\mathcal{M}}(\hat{\mathcal{O}}, Y) := \min_{\hat{o}} D_{\mathcal{M}}(\hat{o}, Y)$.

Let $\hat{\mathcal{O}}_X$ denote the object in state X. For a measurement Y_k, define the *measurement error* to be

$$u_k(X) := D_{\mathcal{M}}(\hat{\mathcal{O}}_X, Y_k). \tag{8}$$

Then the measurement potential is computed as

$$v_k(X, Y_k) := \frac{1}{2}u_k^2(X). \tag{9}$$

Similarly to total measurement energy, we also define the *total measurement error* to be

$$u(X) := \sqrt{\sum_k D^2_{\mathcal{M}}(\hat{\mathcal{O}}_X, Y_k)}. \tag{10}$$

Then, we can re-write π as

$$\pi(X) = \exp\left(-\frac{1}{2}u^2(X)\right). \tag{11}$$

While early Bayesian tactile localization work used sampled measurement models [6], [7], the model described here is analytical. Hence it can be computed efficiently on the fly and without the need for prior training. As all proximity models, the model assumes that the closest point on the object caused the measurement. This is often referred to as a *hard assignment* meaning that the point causing the measurement is assigned to be the closest point. Alternatively with a *soft assignment*, one considers the contribution from all points to the probability of the measurement. Although the soft assignment model has been used for tactile object localization [30], we specifically chose to use the hard assignment model for two reasons. First, the hard assignment model can be efficiently computed explicitly unlike the soft assignment model. Second, for an unbiased application of the soft assignment model, one needs to compute a prior over all surface points, i.e. how likely each surface point is to cause a measurement. However, this prior is usually non-uniform and highly dependent on the object shape, the manipulator shape, and the probing motions.

Like all proximity models, the model described here does not take negative information into account. In other words it does not incorporate information that the robot was able to move through some parts of space without making contact

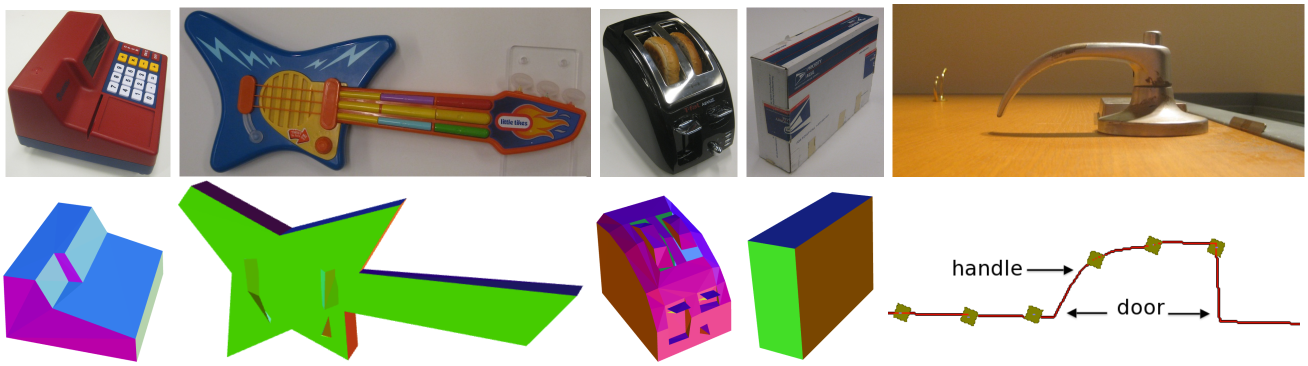


Fig. 2. The five objects used in our experiments: cash register, toy guitar, toaster, box, and door handle. Bottom row shows polygonal mesh models of the objects. Model complexity ranges from 6 faces (for the box) to over 100 faces (for the toaster).

with the object. Negative information has been taken into account in [8] and [30]. However, incorporation of negative information leads to more complex measurement models and complicates inference. The proposed model is continuous and almost everywhere differentiable. Both of these properties would be lost with incorporation of negative information. Although we did not see a significant impact of negative information on accuracy and reliability of localization, it can be useful for active exploration strategies as in [8]. In these cases, the negative information can be superimposed on top of the posterior computed using the proximity model.

D. Motion Model

Since free standing objects can move during probing, we need to define a motion model for this dynamic process. We assume the state of the object evolves via addition of Gaussian noise. Hence, $p(X_t|X_{t-1})$ is a Gaussian with mean at X_{t-1} and variances σ^2_{met} and σ^2_{ang} along metric and angular axes respectively.

IV. Inference Algorithm

We start by introducing the required concepts and providing an intuitive description of the algorithm. A formal description is given in Sect. IV-B. Sects. IV-C through IV-F provide detailed analysis of the algorithm's features and properties.

A. Concepts and Intuition

As we have seen in Sect. II prior approaches have struggled to solve the full 6DOF object localization problem with global uncertainty. The main challenge is computational complexity, which is proportional to the number of particles used. As we will see below, the number of particles required to solve the problem reliably is exponential in the dimensionality of the problem.

1) Required number of particles: As an example consider a 1-D space [0,1]. We want to find the peak of the posterior by sampling particles from the space randomly[4] (see Fig. 3 top left). When we sample a particle from the entire space, the probability of it hitting the support of the peak is equal to the ratio between the width of the peak and the width of the entire space. Let's denote this ratio by $1/\rho$. Hence in expectation we need to sample ρ particles from the entire space in order to get a particle from the support of the peak.

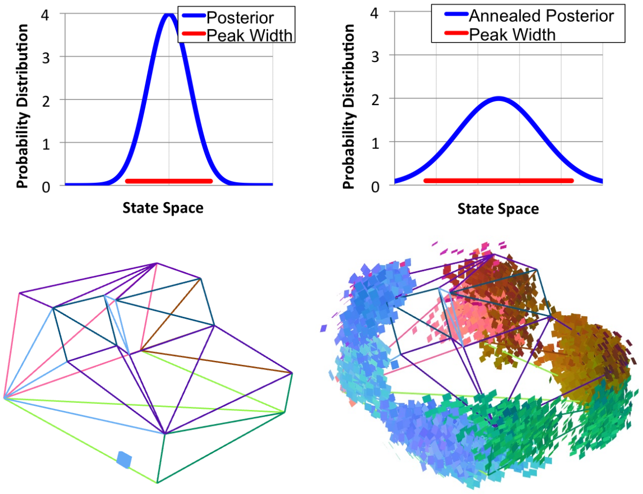


Fig. 3. Top row: two plots of a simple posterior over $[0, 1]$. Top left: true posterior. Top right: annealed posterior. Note that annealing increases peak width, and therefore improves the ratio of peak width to space width. Bottom row: true (left) and annealed (right) posterior for localization of cash register. The cash register model is shown as a wire frame. The small colored squares represent high likelihood particles. Note that annealing makes the problem more ambiguous.

The same is true for higher dimensional problems: *the ratio between the width of the peak and the width of the initial uncertainty dictates the necessary number of particles required for reliable state estimation.*[5]

Unfortunately ρ goes up exponentially with problem dimensionality. For the 3DOF tactile object localization with 400mm initial uncertainty[6] and sensor accuracy of 1mm, ρ comes out to be around 6×10^6, whereas for the 6DOF problem it is approximately 3×10^{15}. To put the exponential blowup in perspective, if we assume that the 3DOF problem takes 1 second to solve, then the 6DOF problem would take approximately 1.5 years.

Thinking in terms of peak width also helps understand the following surprising fact about posterior estimation: *the*

[4] More precisely: we want to sample the particles uniformly and independently.

[5] Of course, in 2-D the term "width" should be replaced by "area", and in 3-D and higher by "volume" of the supporting regions.

[6] Unrestricted orientation uncertainty is assumed.

problem actually becomes harder with more accurate sensors. The reason is simple, more accurate sensors produce more narrow peaks, and therefore ρ increases. In the extreme, when the sensors are perfectly accurate, most Bayesian methods break mathematically.

2) Smoothing: In order to improve the peak width to uncertainty ratio, many modern methods utilize smoothing (also known as relaxation) [39]. Smoothing broadens the peaks (Fig. 3 top right), and therefore reduces the number of particles required to find it reliably. One of the most common smoothing techniques is annealing, which is obtained by exponentiating the measurement model to the power $1/\tau$, where τ is the temperature. Thus for $\tau = 1$ the true measurement model is obtained and for $\tau > 1$ the measurement model is "heated-up". The higher the temperature, τ, the broader the peaks. However, annealing (and any other type of smoothing) comes at a price. The estimates become less accurate and the state estimation becomes more ambiguous (Fig. 3 bottom row). Intuitively smoothing is analogous to blurred vision: the more blurry the vision, the harder it is to determine an object's position or to disambiguate objects.

3) Broad particles: Estimation of a posterior by particles would be impossible without some sort of local smoothness. Indeed, if the value of the posterior at one point was completely unrelated to its value at the neighboring points, then no number of particles would be sufficient for accurate estimation. Most particle based methods do not make this assumption explicit and define each particle as a single point. However, we use broad particles, which represent regions of space around them. We will call them *δ-neighborhoods*[7], where δ is the radius[8] of the neighborhood. Of course the value of δ depends directly on the smoothness of the posterior: the smoother the posterior the larger the δ. "Heating-up" the measurement model increases δ as it makes the posterior smoother. Thus δ depends on the temperature during annealing.

4) Intuitive algorithm description: The main idea is to have the whole uncertainty region covered with δ-neighborhoods. This way we are sure that we have a good approximation of the posterior. At high temperature this can be easily done with just a few particles because δ is large. Of course this will not produce accurate estimates, so we use an iterative refinement approach. First we solve the problem with a few very broad particles at high temperature. Prune out the low probability regions and keep the peaks. Then refine the estimates at a lower temperature. Prune again and repeat until the temperature reaches $\tau = 1$. This way the final estimates will be as accurate as the data and the model allow.

Both the uncertainty region and the peak width change during refinements. The uncertainty region changes due to pruning. The peak width changes due to annealing. Therefore the ratio of peak width to uncertainty width also changes. Hence no single fixed number of particles will work well for all refinement stages. Instead of using a fixed number of particles, we specify the desired particle density by setting the number of particles to maintain per δ-neighborhood. This way the algorithm can compute the appropriate number of particles to use at each refinement stage.

B. The Scaling Series Algorithm

The goal of the algorithm is to compute an approximation of the posterior bel by weighted particles. The initial uncertainty is assumed to be uniform over the starting region. In this case, the posterior is proportional to the data probability (see Sect. III-A1). Hence the weights can be computed via π.

The formal algorithm listing is given in Alg. 1. The algorithm takes as input the initial uncertainty region, V_0, the data set, $\mathcal{D}$, and two user-specified parameters: M and δ_*. M specifies the number of particles to maintain per δ-neighborhood. δ_* specifies the terminal value of δ. The refinements stop once the algorithm reaches this value. Selection of appropriate values for the two user-specified parameters is discussed in Sect. IV-C.

Lines 1 – 3 set initial values. δ_0 is selected so that one δ_0-neighborhood contains the entire initial uncertainty region. The scaling factor $zoom$ is set so that the volume of each δ-neighborhood is halved during scaling. The number of iterations N is computed based on the ratio of initial to final volume. S_δ denotes a δ-neighborhood, $\mathcal{R}(\cdot)$ denotes the radius and $Vol(\cdot)$ denotes the volume of a region.

The initialization is followed by a loop that performs the refinement iterations in lines 4 – 11. At each iteration n, δ_n is computed by applying the scaling factor to δ_{n-1}. The corresponding temperature, τ_n, is computed based on the assumption that δ_* corresponds to the temperature of $\tau = 1$. Line 7 draws a particle set $\bar{\mathcal{X}}_n$ uniformly from V_{n-1} ensuring the required density of M particles per δ-neighborhood. A listing of this procedure is provided in Alg. 2. In line 8 Compute_Normalized_Weights procedure weighs the particles by the annealed data probability, $\pi(X)^{1/\tau_n}$, at temperature τ_n. This procedure also normalizes the weights so that they add up to 1. Line 9 prunes low probability regions. A detailed discussion of this step is provided in Sect. IV-C. Line 10 computes the resulting subregion V_n for this iteration. After completion of the refinement steps, lines 12 and 13 draw the final particle set and compute weights at temperature $\tau = 1$.

The algorithm returns an approximation of the posterior represented by a weighted particle set $\mathcal{X}$, where the weights $\mathcal{W}$ are set to the data probability at temperature $\tau = 1$.

For line 7, we need a procedure to sample uniformly from V_{n-1}, which is represented as a union of δ-neighborhoods. During sampling we need to ensure that we draw M particles from each δ-neighborhood. Thus in effect this is very similar to stratified sampling, except the sets comprising V_{n-1} are not necessarily disjoint. One of the simplest implementations is based on rejection sampling (Alg. 2)[9].

C. Discussion of Algorithm Features and Settings

1) Even density cover: Although this is one of the most crucial features of Scaling Series, at first it may seem counter-intuitive to call the Even_Density_Cover procedure (line 7 of

[7] In earlier versions of the paper, δ-neighborhoods were called δ-spheres.

[8] Radius of a region can be defined as half the diameter, where the diameter is the largest distance between two points contained in the region.

[9] A historical note: the original implementation of this step was more complicated. The use of rejection sampling for this purpose was proposed by an anonymous reviewer at ICRA 2006.

```
Input: $V_0$ - initial uncertainty region, $\mathcal{D}$ - data set, $M$ - number of particles per $\delta$-neighborhood, $\delta_*$ - terminal value of $\delta$.
1:  $\delta_0 \leftarrow \mathcal{R}(V_0)$
2:  $zoom \leftarrow 2^{-1/dimX}$
3:  $N \leftarrow log_2(Vol(S_{\delta_0})/Vol(S_{\delta_*}))$
4:  for $n = 1$ to $N$ do
5:      $\delta_n \leftarrow zoom \cdot \delta_{n-1}$
6:      $\tau_n \leftarrow (\delta_n/\delta_*)^2$
7:      $\bar{\mathcal{X}}_n \leftarrow$ Even_Density_Cover($V_{t-1}, M$)
8:      $\mathcal{W}_n \leftarrow$ Compute_Normalized_Weights($\bar{\mathcal{X}}_n, \tau_n, \mathcal{D}$)
9:      $\mathcal{X}_n \leftarrow$ Prune($\bar{\mathcal{X}}_n$)
10:     $V_n \leftarrow$ Union_Delta_Neighborhoods($\mathcal{X}_n, \delta_n$)
11: end for
12: $\mathcal{X} \leftarrow$ Even_Density_Cover($V_N, M$)
13: $\mathcal{W} \leftarrow$ Compute_Normalized_Weights($\mathcal{X}, 1, \mathcal{D}$)
Output: $(\mathcal{X}, \mathcal{W})$ - a weighted particle set approximating the posterior.
```

Alg. 1: Scaling Series algorithm for posterior estimation.

```
Input: $V$ - sampling region represented as a union of $\delta$-neighborhoods $\{S_i\}$, $M$ - number of particles to sample per $\delta$-neighborhood.
$\mathcal{X} \leftarrow \{\}$
for $i = 1$ to $|\{S_i\}|$ do
    for $m = 1$ to $M$ do
        sample a point $X$ from $S_i$
        reject $X$ if it is in $S_1 \cup \ldots \cup S_{i-1}$
        otherwise add $X$ to $\mathcal{X}$
    end for
end for
Output: $\mathcal{X}$ - a set of particles that evenly cover $V$.
```

Alg. 2: Even_Density_Cover: procedure for uniform sampling from a region represented as a union of δ-neighborhoods with density M per δ-neighborhood.

Alg. 1). Indeed the particle set comprising V_{n-1} is already weighted by the annealed data probability. Why not perform a weighted resample? The weights already resemble the posterior distribution, so why should we discard them and sample particles uniformly instead?

It turns out this step is critical for reliable handling of multi-modal posteriors. This is easiest to understand by considering a simple example. Suppose we have a posterior with two modes of even height. We draw two particles: one near each mode. If one of the particles is slightly closer to a mode than the other, the weights will be uneven. Hence during weighted resampling we will favor one mode over the other. If we perform several iterations, this error compounds and hence we are quite likely to discard one of the two modes. Even_Density_Cover avoids this problem. If a particle survived the pruning step, it will be given full consideration at the next iteration.

The multi-modal case is important for two reasons. First, multi-modal posteriors arise naturally during tactile object exploration because at early stages the number of measurements is insufficient to determine the object's location unambiguously. In fact the posterior can even have entire regions of high probability (see Fig. 4). Estimating the multi-modal posterior at early stages of exploration is important for making safe and informed decisions about future sensing actions. Second, note that most iterations of Alg. 1 compute the annealed posterior. The higher the temperature the more ambiguous the posterior becomes (see bottom row of Fig. 3). Hence multiple modes are often present during early iterations of Scaling Series as we show in Sect. V-E1 experimentally.

One other important reason for the Even_Density_Cover step is that without it we would be double-counting the data and hence the estimate would not converge to the true posterior.

2) Pruning: The purpose of the pruning step (line 9 of Alg. 1) is to remove low probability regions from consideration. This way the computational resources can be focused on the more interesting high probability regions. This step removes particles with relatively low weights from the particle set. This is achieved via weighted resampling. See [53] for a listing of a weighted resampling algorithm. During this step the value of M is ignored. Instead this procedure draws the same number of particles as there were prior to this step. The weights are set to be uniform after the resampling operation. Although weighted resampling is likely to discard low probability particles, from theoretical viewpoint the resulting particle set encodes the same probability distribution as the weighted particle set prior to resampling.

3) Selecting δ_*: The value for δ_* should be selected so that the posterior changes only a small amount within a δ-neighborhood of any particle. This can be done using the *Lipschitz constant*.[10] For the global localization case, the posterior is proportional to $\pi(X) = \exp(-\frac{1}{2}u^2(X))$ (see Eqn. 11), so we set

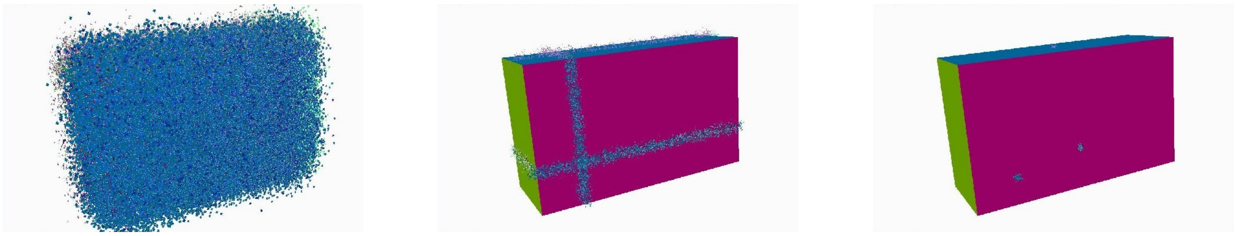
$$\delta_* := \frac{1}{\lambda_\pi}, \tag{12}$$

where λ_π is the Lipschitz constant of π. It can be easily shown that λ_π is bounded by $\lambda_u/\sqrt{e}$ (see Appendix A), and so Eqn. 12 relates δ_* to λ_u. Thinking in terms of u gives δ_* a physical meaning: it is the largest radius, within which the total measurement error can change by at most $\sqrt{e}$. Lipschitz constant computations for the measurement model described in Sect. III-C are provided in Appendix A.[11]

The measurement model described in Sect. III-C is continuous with bounded derivatives almost everywhere. Thus it is guaranteed to have a Lipschitz constant. However, for some measurement models the Lipschitz constant may not exist or be cumbersome to compute. In these cases, one can set the value of δ_* to a good guess, which works well in most areas of state space. Increasing the value of M will help compensate for an imperfect setting of δ_*, as these two parameters complement each other.

[10] For a function $f(X)$, the *Lipschitz constant*, λ_f, is defined to be the maximum slope between any two points.

[11] These derivations provide upper bounds that hold for all objects and data sets. Although these values are not necessarily optimal for a specific object and data set, they serve as a good guide and can be further optimized experimentally (see Sect. V-E4).



(a) after 1 measurement

(b) after 2 measurements

(c) after 3 measurements

Fig. 4. During exploration the posterior evolves as additional measurements arrive. The particles in this figure approximate high likelihood regions of the evolving posterior. Each particle is shown by a small square at the hypothesized position of the first data point on the surface of the object. The normal of each square corresponds to the sensed normal transformed to the object coordinate frame based on the hypothesized object pose.

4) Shape of δ-neighborhood: So far we have not specified what shape a δ-neighborhood takes. In early versions of the paper [9], we termed the neighborhoods δ-spheres and defined them to be hyper-spheres of radius δ. However, we also mentioned that when coordinates are not homogenous (e.g. position vs. orientation), scaling factors may be needed. Hence, the obtained shape is actually a hyper-ellipsoid. The scaling factors can have a significant impact on performance. Analogously to using the Lipschitz constant of π, the neighborhood dimensions along each axis can be set based on the partial Lipschitz constants of π, which are defined as the maximum partial derivatives. If $\lambda_{\pi,i} := \sup|\frac{\partial \pi}{\partial x_i}|$, then we set the radius of the neighborhood along i-th axis to be

$$r_i := 1/\lambda_{\pi,i}. \tag{13}$$

In this case we assume that $\delta_* := r_1$ to avoid ambiguity. See Appendix A for a derivation of the partial Lipschitz constants for the model described in Sect. III-C.

5) Annealing schedule: During iterations we compute the annealed data probability: $\pi^{1/\tau}(X) = \exp(-u^2(X)/\tau)$, and so τ acts on $u^2(X)$. Since δ_* is proportional to change in $u(X)$, τ should be adjusted in proportion to δ^2 rather than linearly with δ. This computation takes place in line 6 of Alg. 1.

6) Selecting M: The number of particles to maintain per δ-neighborhood is a user-specified parameter, which affects reliability, efficiency, and accuracy. As we already mentioned it complements the value of δ_*. The higher the value of M, the higher the accuracy and reliability, and also the higher the computational cost. In practice, if δ_* is chosen as described above, M values between 3 and 6 tend to give good results, although in rare cases M can be set as low as 2. A higher value of M is needed if the Lipschitz constants have been underestimated or if these constants do not exist. An empirical evaluation of dependence on M is provided in Sect. V-E4.

7) Comparison to APF: At a first glance Scaling Series may seem very similar to APF, which also uses iterative annealing. However there are three important distinctions. First, while APF has a fixed number of particles to use at each iteration, Scaling Series selects the number of particles automatically and dynamically for each refinement stage. The selection takes into account the smoothness of the posterior, the total uncertainty volume, and the width of the neighborhood each particle can represent. Thus, the optimal number of particles is used at each iteration for efficient and accurate representation.

Second, while APF is known to handle poorly in multimodal scenarios [50], Scaling Series handles these very well due to the use of Even_Density_Cover. For this reason APF does not converge to the true posterior, whereas Scaling Series does as we show in Sect. IV-F.

Third, the Scaling Series annealing schedule is derived from the mathematical properties of the posterior. This allows for much more efficient and straightforward annealing than APF, which relies on survival rate. Scaling Series also derives the relationship between temperature and δ, which is analogous to the APF diffusion rate. In APF, the diffusion rate is disassociated from the temperature, which can lead to non-optimal diffusion. Empirical comparison to APF is provided in Sects. V-E2, V-E3, and V-D.

D. Algorithm Variations

1) Zoom factor: The standard version of Scaling Series algorithm sets $zoom$, so that the volume of a δ-neighborhood is halved at each iteration. However, it is possible to zoom faster or slower, reducing the volume for example to 10% or 90% each time. Note that if $zoom$ factor is changed, the number of iterations also needs to be changed in line 3, where the base of the $\log$ is the factor, by which the volume is reduced per iteration. Faster zooming will require fewer iterations, slower will require more. Empirical evaluation in Sect. V-E5 shows that $zoom$ factor of the original algorithm is optimal.

2) Alternative pruning strategies: One alternative strategy for pruning is thresholding based on a preset percentage of the top weight in the particle set. Unlike weighted resampling, thresholding can be carried out based on log of the weights (i.e. directly on v). This can significantly improve numerical stability in situations where the data does not match the model very well — a common scenario in the presence of unmodeled effects.[12] A threshold of ξ corresponds to $\ln \xi$ in terms of v. Thus any particle whose v exceeds the minimum v in the particle set by more than $\ln \xi$ can be pruned. A reasonable choice is to prune out everything that is further than one standard deviation away from the solution. Since π is Gaussian in u, this results in $\xi = \frac{1}{\sqrt{e}} \approx 60\%$. Empirical evaluation in Sect. V-E6 shows that this is indeed the optimal setting.

[12] Although thresholding in log space does not change the mathematical outcome of the operation in principle, in practice when the probability of the data is extremely low, the weights come out to be zero due to limited floating point exponent range.

3) Time limit: One practical approach is to limit the amount of time allotted for estimation based on a single data set. This is especially helpful at early stages of exploration, when the posterior is highly ambiguous (Fig. 4). See Sect. V-B2 for an example.

4) Compensating for object symmetries: Many man-made objects have symmetries that can not be resolved no matter how much data is collected. These objects always produce multi-modal posteriors. In order to reduce the number of particles to represent the modes, a simple strategy is to take each state X modulo the symmetries.

E. Tracking Dynamic Objects

So far we have only considered estimation of posteriors with a uniform prior. This works well for stationary objects. However, free standing objects can shift during tactile probing. Hence we need a method for tracking the state of dynamic objects. In these cases the prior is not uniform as it encodes the information from prior sensing actions and possible motions of the object. Hence we need a way to extend Scaling Series to tracking of dynamic objects.

First let us consider how a standard particle filter (PF) solves this problem (see [39] for details). At each time step t, PF performs a motion update followed by a measurement update. The motion update performs a resample followed by application of the motion model. The measurement update incorporates the most recent data by setting importance weights proportional to π_t. Note, that the measurement update is similar to Scaling Series, except the prior is non-uniform in this case.

We consider three possible ways of extending Scaling Series to the tracking problem. The first algorithm, SS-DYN1, simply runs Scaling Series during the first time step (when the prior is uniform), and then follows by standard particle filter updates for the rest of the time steps.

The second algorithm, SS-DYN2, is the same as SS-DYN1, except that it uses Scaling Series during each measurement update. To do so, it uses the particle set generated by the motion update of the previous step and sets δ_0 broad enough to encompass motion noise. Of course, this does not fully take the prior into account, so we end up "forgetting" some information from prior time steps.

The third algorithm, SS-DYN3, runs Scaling Series on each data set using a uniform prior, and then adjusts the weights to capture the motion model via the Bayesian recursion equation (Eqn. 6). This way it does not "forget" any information from prior steps. Formal listing of SS-DYN3 is provided in Alg. 3. The algorithm takes as input the posterior from the prior time step represented as a set of weighted particles. The rest of the parameters are analogous to Alg. 1. In line 1 the algorithm approximates π_t with a set of weighted particles using Scaling Series (Alg. 1). Lines 4 – 6 compute the integral that appears in Eqn. 6. Line 7 multiplies the weights by the integral. The weights are then normalized in line 9. The algorithm outputs the resulting weighted particle set, which approximates the posterior at time step t.

Note that due to efficiency of Scaling Series, SS-DYN3 algorithm is tractable as is. However, two efficiency improvements can be implemented. First, if the prior state is too far away from the proposed current state, the probability of the object transitioning from one state to the other is very low. Thus the contribution of this term to the integral in line 5 is negligible. Hence, the loop in lines 4 – 6 can be restricted to particles X_{t-1} that are close enough to X_t.

```
Input: $(\mathcal{X}_{t-1}, \mathcal{W}_{t-1})$ - weighted particle set from prior time
      step, $V_t$ - initial uncertainty region, $\mathcal{D}_t$ - data set for time
      step $t$, $M$ - number of particles per $\delta$-neighborhood, $\delta_*$ -
      terminal value of $\delta$.
1: $(\mathcal{X}_t, \mathcal{W}_t) \leftarrow$ Scaling_Series$(V_t, \mathcal{D}_t, M, \delta_*)$
2: for each $(X_t, w_t) \in (\mathcal{X}_t, \mathcal{W}_t)$ do
3:     $s \leftarrow 0$
4:     for each $(X_{t-1}, w_{t-1}) \in (\mathcal{X}_{t-1}, \mathcal{W}_{t-1})$ do
5:         $s \leftarrow s + p(X_t|X_{t-1})w_{t-1}$
6:     end for
7:     $w_t \leftarrow w_t s$
8: end for
9: normalize weights $\mathcal{W}_t$
Output: $(\mathcal{X}_t, \mathcal{W}_t)$ - a weighted particle set approximating
      the posterior at time step $t$.
```

Alg. 3: SS-DYN3: algorithm for tracking a dynamic state with Scaling Series.

Second, we can initialize V_t to the high probability regions of the prior $\overline{bel}_t$. In other words, we can focus on areas, where the object is likely to move based on prior information. We can compute the prior using the motion update step of standard particle filters. Hence we perform a weighed resampling from $(\mathcal{X}_{t-1}, \mathcal{W}_{t-1})$ followed by application of the motion model with randomly sampled noise parameters. The result is an unweighted particle set representing $\overline{bel}_t$. Then V_t can be set to the union of δ-neighborhoods centered at the obtained particles, where δ should be set broad enough to accommodate for the error due to having a finite number of particles. In line 1 of Scaling Series, δ_0 should be set to the value of δ used for V_t. Note that this efficiency improvement does not double-count the prior, due to the Even_Density_Cover step at the beginning of Scaling Series.

F. Algorithm Analysis

In this section we analyze convergence of the proposed algorithms. In short, we show that Scaling Series, SS-DYN1 and SS-DYN3 converge to the true posterior. However, SS-DYN2 does not converge.

Scaling Series estimates of the posterior converge as M tends to ∞. The convergence is understood in the same sense as for importance sampling. Namely, we want to estimate the expected value $\mathbb{E}[f; bel]$ of some function of interest $f(X)$ with respect to the posterior distribution. Let the estimate produced by importance sampling be denoted by

$$\mathrm{IS}_J(f) := \sum_j f(X_j) w_j, \tag{14}$$

where X_j are particles and w_j are normalized importance weights. Then we know that $\mathrm{IS}_J(f) \to \mathbb{E}[f; bel]$ *almost surely* (a.s.) as $J \to \infty$ [54].

Similarly, let the estimate produced by Scaling Series with M particles per δ-neighborhood be denoted by

$$\mathrm{SS}_M(f) := \sum_j f(X_j) w_j, \tag{15}$$

where X_j are particles in the final set $\mathcal{X}$ with normalized weights w_j. Then analogously to importance sampling we have the following convergence result.

Theorem 1: $\mathrm{SS}_M \to \mathbb{E}[f; bel]$ *a.s. as* $M \to \infty$.

Proof: Let us consider the first iteration of Scaling Series (Alg. 1). Particles in $\bar{\mathcal{X}}_1$ a.s. completely cover V_0 as $M \to \infty$, and so particles in $\mathcal{X}_1$ also a.s. completely[13] cover V_0. The same reasoning can be applied to all N iterations. Hence, V_N a.s. completely covers V_0 because N does not depend on M. When V_N covers V_0, lines 12 & 13 of Scaling Series are equivalent to importance sampling with a uniform prior and with $J > M$ particles. Thus by convergence of importance sampling we get the desired convergence result for Scaling Series. □

Similarly we can derive convergence of SS-DYN1 and SS-DYN3 from the convergence of particle filters (PF) [55]. However, SS-DYN2 does not converge to the true posterior because the Even_Density_Cover step after propagating the particles discards some information from the prior, and SS-DYN2 does not compensate for this information loss. This is similar to the behavior of APF, which also does not converge to the true posterior due to information loss caused by annealing. In practice, however, these algorithms can be very useful (as we show in Sect. V-D).

Theorem 2: SS-DYN1$_M$ *converges a.s. as* $M \to \infty$.

Proof: SS-DYN1 consists of SS followed by PF, so the result follows from their convergence. □

Theorem 3: SS-DYN3$_M$ *converges a.s. as* $M \to \infty$.

Proof: In SS-DYN3, line 1 computes π_t and lines 4 – 6 compute the prior $\overline{bel}_t$ using Eqn. 5. Line 7 multiplies the weights by the prior, and hence by Bayesian recursion the resulting weights are proportional to the posterior bel_t. □

V. Experimental Results

We performed extensive evaluation of Scaling Series with both real and simulated data. Two implementations were used: the old and the new one. The old implementation was in Java running on a 1.2GHz laptop computer. The new implementation is in C++ running on a 2GHz laptop computer.

We constructed polygonal mesh models of five everyday objects: cash register, guitar, toaster, box, and door handle (Fig. 2). The mesh models of the first three objects were constructed based on measurements taken with the robot's end effector. Models for the last two objects were constructed from ruler measurements. The accuracy of models ranges from 5mm for the first three objects to 1mm for the last two objects. Accuracy of surface normals is quite poor near edges, corners, and other non-flat parts of the objects.

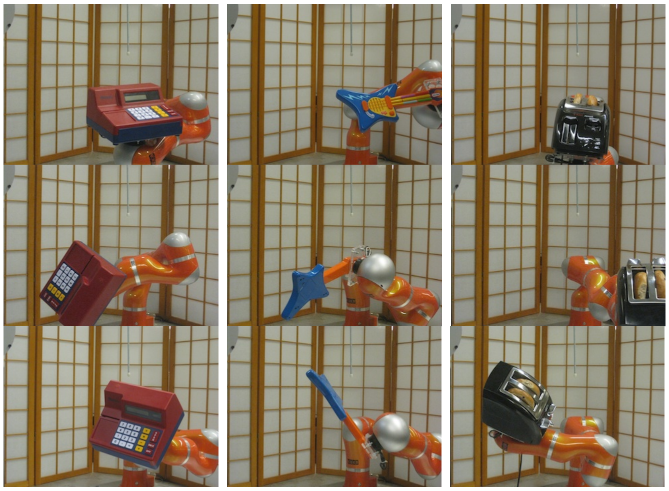


Fig. 5. The nine poses used during localization experiments (three poses per object). These poses where selected randomly from the uncertainty region. The poses in the top row were also used to run object manipulation scenarios (see videos). All pictures were taken from the same vantage point.

Each object model included feature points: buttons, levers, grasp points, etc. Once localization is performed, the features are transformed into robot coordinates so that the manipulation scenarios could be carried out. Videos of the experiments, code and other supplemental materials are available on our website [56].

The remainder of this section is organized as follows. Sects. V-A, V-B, and V-C cover real robot experiments with the five objects mentioned above. Sects. V-D and V-E cover experiments performed in simulation. Sect. V-D considers tracking of a free standing box that moves during tactile exploration. Sect. V-E provides extensive empirical evaluation of Scaling Series features and parameters.

A. Experiments with Cash Register, Guitar, and Toaster

In this set of experiments we evaluated the algorithm on three common objects: cash register, guitar, and toaster. The manipulator used was a 6DOF PUMA robot, equipped with a 6-D JR3 force/torque sensor at the wrist. In these experiments we used a long end effector of 300mm length and 6mm diameter. Since the initial uncertainty was large, the long end effector was necessary to ensure that the robot always made contact with the tip of the end effector and not some other non-sensing part of the robot. The end effector had a semi-spherical tip of 5mm radius.

The sources of error included: mesh model inaccuracies, object deformation (especially noticeable for the guitar), robot positioning error, end effector deformation (significant due to the long length), and error due to unknown position of the contact on the tip of the end effector. Although it is difficult to determine exact amount of noise produced by all of these errors, we estimated the contact position noise to be roughly $\sigma_{\text{pos}} = 5\text{mm}$. Sensed normals were extremely noisy due to polygonal model inaccuracy and long end effector length. We used $\sigma_{\text{nor}} = 50°$. The experiments were carried out using our C++ implementation of Scaling Series with thresholding. We set the threshold to $\xi = 60\%$ and used $M = 6$ particles per neighborhood. The rest of the parameters were set in

[13] This statement is true as long as $bel(X) > 0$ for all X. Without loss of generality we can assume that this is the case. Otherwise, we can simply exclude from V_0 points at which $bel(X) = 0$ as these points do not contribute to the expectation of f.

TABLE I
Results of the nine experiments with cash register, guitar, and toaster. Cartesian coordinates and errors are listed in millimeters. Orientation angles and errors are listed in degrees. Localization errors are reported with respect to ground truth poses obtained from the Kuka robot.

		Pose						No.	Data	xyz	$\alpha\beta\gamma$	
No.	Object	x	y	z	α	β	γ	Probes	Points	Error	Error	δ_*
1	Register	393	542	-285	-60°	22°	16°	24	12	2.9	3.5°	2.4
2	Register	131	635	-350	128°	67°	41°	24	11	6.8	4.3°	2.5
3	Register	364	520	-275	-18°	8°	-33°	26	14	2.1	1.7°	2.2
4	Guitar	468	500	-255	-34°	-32°	-30°	27	10	5.6	2.3°	2.6
5	Guitar	219	528	-335	-166°	-4°	31°	36	11	9.2	5.2°	2.5
6	Guitar	273	678	-186	76°	70°	-68°	60	17	4.8	3.0°	2.0
7	Toaster	380	445	-310	127°	161°	-11°	20	11	4.2	2.4°	2.5
8	Toaster	576	271	-286	-25°	1°	-7°	23	11	6.1	1.2°	2.5
9	Toaster	180	614	-204	85°	101°	39°	22	14	5.5	3.3°	2.2
	Min	131	271	-350	-166°	-32°	-68°	20	10	2.1	1.2°	2.0
	Average	—	—	—	—	—	—	29	12.3	5.2	3.0°	2.1
	Max	576	678	-186	128°	161°	41°	60	17	9.2	5.2°	2.6

accordance with the derivations in Sect. IV-C and Appendix A. Specifically, we set $\delta_*, r_{\rm pos}$, and $r_{\rm ori}$ so that:

$$\begin{aligned} \delta_* &= \sigma_{\rm pos}\sqrt{e/K}, \\ r_{\rm pos} &= \delta_*, \\ r_{\rm pos}/r_{\rm ori} &= \sqrt{\mathcal{R}^2(\mathcal{O}) + \sigma^2_{\rm pos}/\sigma^2_{\rm nor}}. \end{aligned} \tag{16}$$

The initial uncertainty for all objects was 400mm along x, y, z with unrestricted orientation. We randomly selected nine poses from this uncertainty region: three poses per object (Fig. 5). The objects were held in place by a Kuka LWR robot. We used the joint angles of the Kuka robot to generate ground truth for all nine poses.

Prior to experiments we generated a set of safe probing trajectories, which took joint limits and collisions with the environment into account. During experiments, data collection procedure randomly selected probing trajectories from the pre-generated set. All probing trajectories moved the robot along the direction of the end effector, so that the end effector tip was the first part to make contact. Each probe took approximately 10s. The Scaling Series algorithm was run on all data points collected up to that time step. The algorithm was allowed to compute until the next measurement arrived. Once the algorithm determined that the posterior had a single mode and all particles were within 10mm of each other, the probing procedure stopped and the mean pose was used as the estimated pose. In experiments 1, 4, and 7, the localization procedure was followed by a manipulation scenario: using the cash register, playing the guitar, and toasting bagels respectively. Videos are available on our website [56].

Localization results for the nine experiments are summarized in Tbl. I. Overall, localization was quite accurate: the average localization error was 5.2mm and 3°. We believe this was an important factor in the success of the manipulation scenarios. Localization was the most accurate for the cash register (3.9mm average error) because its shape consists of planar surfaces that are easy to model accurately. The toaster has many curved surfaces, which are more difficult to model and hence the localization error is slightly higher: 5.3mm on average. Localization was the least accurate for the guitar (6.5mm average error) because this object deformed significantly during probing as can be seen in the videos.

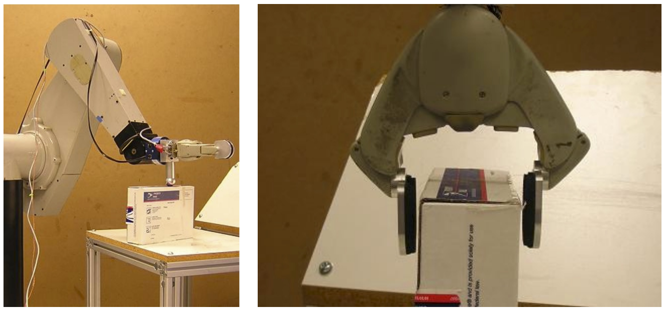
B. Manipulating a Box

In the second set of experiments, we applied the Java implementation of our approach to the task of localizing, grasping and picking up a rectangular box (see Fig. 6). As in the previous set of experiments, we used the PUMA robot with the JR3 sensor. This time the robot's end-effector included a gripper and robotic finger combination, so that the robot could perform both probing and grasping tasks. The finger was much shorter and thicker (75mm length, 25mm diameter) with a spherical end of 15mm radius. This configuration resulted in much more accurate data because the end effector did not deform. Shorter length of the finger also resulted in more accurate measured normals. The rectangular box was 56mm x 159mm x 238mm in size. The size of the mesh model was inflated by the radius of the spherical end-effector, so that the end-effector tip could be reduced to a single point in computations. Due to higher accuracy of the measurements, we set $\sigma_{\rm pos} = 1$mm and $\sigma_{\rm nor} = 5°$. The rest of the parameters were set as before.

We used the same initial uncertainty region: 400mm in x, y, z with unrestricted orientation. This time the probe was too short to safely explore the large uncertainty region without touching the object with non-sensing surfaces. Therefore for data collection we developed a custom active sensing procedure specific to the box object. Although the procedure restricted the set of poses in which the data collection could be successful, localization was still performed on the full uncertainty region without taking the restrictions into account. The box was fixed using brackets so that it remained relatively stationary during the experiments, although it still shifted and deformed during probing as can be seen in the videos.

1) Fully-constrained case: In fully-constrained experiments, we collected five measurements using the above probing procedure. These five points were used to perform localization of the box using Scaling Series. Two grasp points were manually defined on the box model, each consisting of 3 points: one for each side of the gripper and one for the wrist position. Thus each grasp point fully defined position and orientation of the gripper. After localization, the grasp point with the highest z-coordinate was selected[14]. The gripper

[14] z-coordinates increase vertically upwards



(a) sensing (b) grasping

Fig. 6. The stages of the box manipulation experiment. (a) Sensing the box with a robotic finger. (b) Grasping the box. The position and orientation of the box were estimated from the data obtained during sensing stage. The grasping configuration is defined as part of the box model. Note the precise fit required to perform the grasp. The last stage (not shown) is manipulation of the box.

orientation, position and approach vector were derived from the selected grasp point and estimated parameters. Note the precise fit required for grasping in Fig. 6.

We performed 30 trials of fully-constrained experiments on the real robot. The sensing procedure took 30 seconds. Localization was performed in less than 1 second. Out of the 30 trials, the data collection procedure failed in 9 trials[15]. These trials were aborted. In all of the remaining 21 trials, the robot successfully localized, grasped, and manipulated the box.

2) Early stages of exploration: To evaluate the algorithm performance at early stages of exploration, we took data sets consisting of 2 - 3 measurements from different sides of the box. These data sets do not fully constrain the problem, and so the modes of the resulting posterior form ridges in the state space (Fig. 7). For real robot experiments, we took subsets of measurements from our completed real robot trials. We verified that the estimated region included the true state of the object, as it was estimated from complete data sets. We also examined the estimated region visually to make sure it corresponded to the correct solution region in each under-constrained scenario. In addition, we performed 100 simulated trials where ground truth was available. The true state was included in the resulting solution set in all 100 trials.

Since the number of solutions is infinite, high precision settings result in large numbers of particles. However, it is possible to exit out of iterations early based on a time limit setting as discussed in Sect. IV-D3. For example for a data set consisting of two measurements, Scaling Series generated 4,000 particles for δ =11mm and 29,000 particles for δ =1mm (Fig. 7). The running time increases with the number of particles generated. For our Java implementation, operations with a few thousand particles take a few seconds, but 29,000 particles take 40-50 seconds to process. Thus it is possible to trade off precision of estimation for running time. As more measurements arrive, the solution region shrinks and higher precision can be achieved with fewer particles.

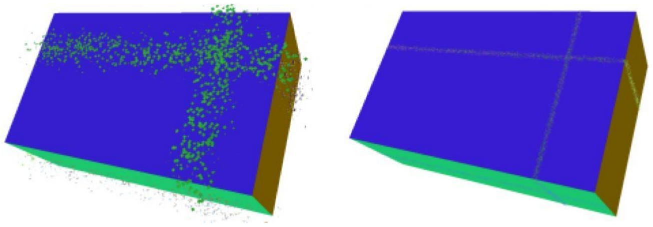
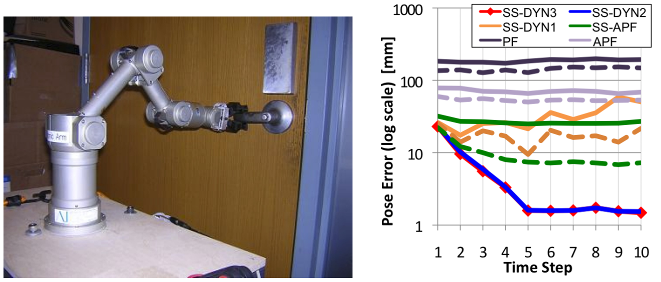


Fig. 7. Examples of under-constrained solution estimation for data sets consisting of 2 measurements (includes symmetry compensation). Left: With δ =11mm, 4,000 particles were generated by Scaling Series. Right: With δ =1mm 29,000 particles were generated. As before each particle is shown by a square indicating the location of the first data point on object surface. The size of each square is δ.



(a) door handle experiments (b) free standing objects

Fig. 8. (a) Harmonic Arm robot operating the door handle in one of the experiments. (b) Accuracy of object tracking over 10 time steps starting with global uncertainty. Each algorithm was given 1s of computation time per step. Dashed lines show how the tracking improves if 60s per step are allotted for algorithms of the corresponding color. The results are averaged over 100 runs.

C. Door Handle Operation

In the third set of real robot experiments, we performed door handle manipulation with a mobile manipulator consisting of a Segway platform and a 5DOF Harmonic Arm 6M manipulator (see Fig. 8(a)). Once the robot navigates to the area in front of a door (using its laser sensors for approximate localization), we use tactile feedback to accurately estimate the position and orientation of the door and the door handle. The Harmonic Arm manipulator used in these experiments has about 1mm end-effector positioning precision. Since all door handles in the building are mounted at the same height and always in horizontal position, the height of the handle as well as two orientation angles were fixed, which reduced the localization task to a 3DOF problem. Our algorithm used a 2-D model of the door that was constructed by hand using ruler measurements. Specifically, we took door handle depth measurements every 10mm along its length in a horizontal plane through the center of the handle. This resulted in a 2-D model consisting of line segments (Fig. 2). The grasping point was defined near the tip of the door handle. The sensing used in this experiment gave only position measurements, and did not include surface normals.

For each experimental trial, the robot took 6 measurements in a 30° span (at $0^\circ, 6^\circ, \ldots, 30^\circ$). Each data point thus consisted of range to the contact point and an orientation angle. The sensing procedure took between 1 and 2 minutes. Using these six measurements, our algorithm was able to localize the door and the door handle in a fraction of a second using our Java implementation. In these experiments, we restricted

[15] During these experiments the PUMA robot was experiencing intermittent sudden jolts possibly due to faulty encoders. These jolts resulted in large force measurements registered on the JR3 sensor and hence were interpreted as phantom contact readings. Trials during which these jolts were experienced were aborted.

the dimensions of the state space (to 60mm x 60mm x 30°) because of the limited operational range of the manipulator. Out of 100 independent trials, our algorithm successfully completed the sensing in 98 trials. In all of these 98 trials, our algorithm then successfully localized, grasped, and turned the door handle, and opened the door. The two failures during sensing were caused by a hardware glitch in communication with the robot.

D. Free Standing Objects

When estimating the state of a dynamic system, it is important that the information gained via measurements exceeds the information lost due to noisy motion at each time step. Otherwise the state will only become more uncertain over time making localization impossible. Since in our hardware setup the robot only has one finger, little information is obtained at each time step placing a very tight restriction on the amount of motion allowed. Hence to evaluate tracking of moving objects, we assume that the robot possesses a multi-fingered hand capable of measuring at least three data points per time step[16]. We evaluated this scenario in simulation by sampling three contacts randomly from the surface of the box using the same box model as in Sect. V-B. The object was tracked over ten time steps, starting with global 6DOF uncertainty (400mm in position, 360° in orientation). We simulated measurement noise of $\sigma_{\rm pos} = 1{\rm mm}$ and $\sigma_{\rm nor} = 5^\circ$ as well as considerable motion noise: $\sigma_{\rm met} = 20{\rm mm}$ and $\sigma_{\rm ang} = 10^\circ$. The rest of the parameters for Scaling Series were set as follows: $\delta_* = 1{\rm mm}$, $r_{\rm pos} = 1{\rm mm}$, $r_{\rm ori} = 1^\circ$, $M = 6$, $\xi = 60\%$.

Using the C++ implementation, we compared SS-DYN1, SS-DYN2, SS-DYN3, two variants of APF, and PF (Fig. 8(b)). The two APF variants were: APF and SS-APF. The standard APF used 100 layers and survival rate $\alpha = 90\%$ with annealing schedule selected as in [48]. These settings performed the best for APF. SS-APF used 20 layers and its annealing schedule was selected using Scaling Series methodology. Hence, SS-APF is in between SS and APF algorithms. It uses the same annealing schedule as SS, but like APF it is missing the Even_Density_Cover step of SS. All algorithms were given 1s of computation time per time step. Dashed lines show how the performance improves with 60s per time step. The results are averaged over 100 runs.

Note, that with three measurements per data set, the posterior is multi-modal during the first several time steps. Hence it is not possible to fully localize the object initially. The ambiguity is gradually resolved as additional measurements arrive. Also, note that the average error is to a large extent a function of reliability. In other words whether or not a particular algorithm found the object at each time step. The lower the reliability, the higher the average error. Thus, even if an algorithm has high average error, it may have accurately localized on some of the runs.

PF was unable to locate and/or track the object as the average error is over 140mm even with 60s per update. Still, the average error improves from 1s to 60s, so with more time per update, PF is likely to perform even better. APF converges to 68mm error, which improves to 53mm if 60s per update are allotted. SS-APF ends up with 27mm average error, which improves to 7mm with 60s per update. SS-DYN1 starts off well due to initialization via SS at first time step, however over time SS-DYN1 diverges and approaches APF error. These results are in line with [50], where APF was compared to well initialized PF.

Both SS-DYN2 and SS-DYN3 performed very well, quickly converging to 1.5mm average error. There was no significant difference in performance of these two algorithms. This is likely because in our case little is known about how the object moves. It is possible that in applications with more informed motion models, SS-DYN3 will show an improvement over SS-DYN2. The difference in performance between SS-DYN3 and SS-APF clearly underscores the importance of the Even_Density_Cover step for estimation of multi-modal posteriors.

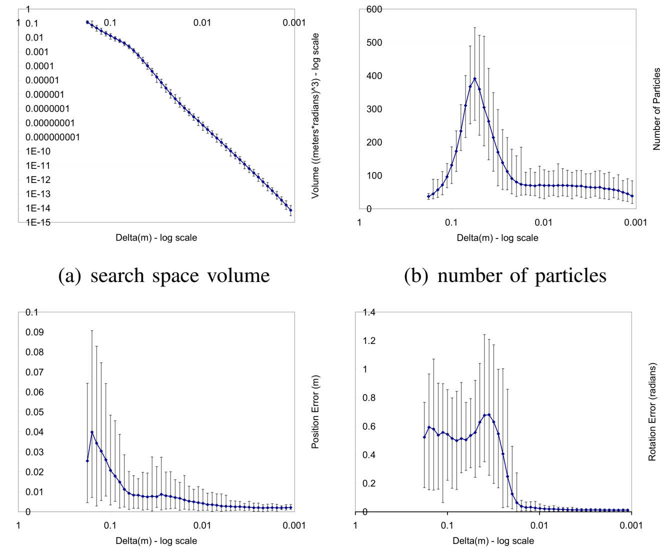
E. Algorithm Evaluation

In this section we evaluate the impact of Scaling Series features and parameters on performance, as well as compare Scaling Series to other algorithms. These experiments are carried out on simulated data for the box localization problem. The same box model was used as in the real data experiments (Sect. V-B). Unless otherwise noted we used the following settings: target resolution $\delta_* = 1{\rm mm}$, hyper-ellipsoid neighborhoods with $r_{\rm pos} = 1{\rm mm}$ and $r_{\rm ori} = 1^\circ$, $M = 6$ particles per neighborhood, $zoom = 1/\sqrt[6]{2}$, measurement noise $\sigma_{\rm pos} = 1{\rm mm}$ and $\sigma_{\rm nor} = 5^\circ$. Fully-constrained data sets (with 5 measurements) were used, unless stated otherwise. Results shown are averages over 100 runs of the algorithms. Most experiments were carried out with our C++ implementation and used Scaling Series with thresholding on $\xi = 60\%$.

In experiments we compare several algorithms, including: Scaling Series (SS), importance sampling (IS), annealed particle filter (APF), and a variant of APF with Scaling Series annealing schedule (SS-APF).

1) Scaling Series evolution over iterations: First we evaluate how the search space and estimation error change during iterations of Scaling Series (Fig. 9). In the plots, the progression of the series is from left to right, with corresponding δ values noted in meters on the horizontal axis (in log scale). The plots clearly show that the volume of the search space shrinks drastically with iterations. At the same time estimation error falls. The number of particles remained small throughout all of the experiments, with the absolute maximum being below 600. The number of particles is highest for δ values between 30 and 100mm. At these settings the distribution is multi-modal, corresponding to 6 possible sides of the box. As these possibilities are ruled out, the number of particles goes down. The multi-modality is particularly noticeable on

[16] One possibility is a hand with three fingers, each consisting of three phalanges with a tactile sensor on each phalange. Thus in principle this hand is capable of making nine contacts during a single grasp of the object. If the hand is operated compliantly (either in software or hardware), then it can close around the object without knowing its exact location. As it closes it will make multiple contacts. Since the blind grasp may not be very good, we assume that only three out of possible nine contacts are sensed.



(a) search space volume

(b) number of particles

(c) position error

(d) orientation error

Fig. 9. Performance of Scaling Series on simulated data set during 100 experiments. Each graph shows progression of the series from left to right. Corresponding value of δ is noted on the horizontal axis in meters, log scale. Vertical bars represent absolute min/max values during all 100 runs.

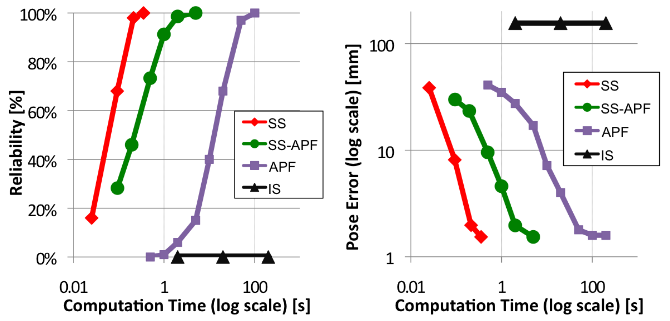
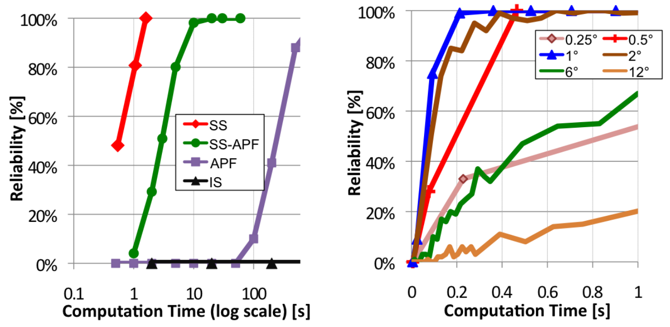


Fig. 10. Comparison of SS, SS-APF, APF, and IS on single mode posteriors. Left: reliability vs. computation time. Right: accuracy vs. computation time.



(a) multi-modal

(b) neighborhood shape

Fig. 11. (a) Comparison of SS, SS-APF, APF, and IS on multi-modal posteriors. The plot shows percentage of successful runs, in which each algorithm found all modes. (b) Impact of changing δ-neighborhood shape on reliability of Scaling Series. Hyper-ellipse radius along orientation angles, r_{ori}, was changed during these experiments, while we kept $\delta_* = r_{\text{pos}} = 1\text{mm}$. The legend shows r_{ori} values in degrees. Computation time was varied by changing M.

the orientation error plot (Fig. 9(d)). These experiments used our Java implementation.

2) Single mode estimation: In this set of experiments we used fully-constraining data sets (5 measurements from different sides of the box), so that the resulting posterior was unimodal. We compared reliability and accuracy of SS, IS, APF, and SS-APF (Fig. 10). For SS-APF, we show performance with 20 layers, which worked the best. For APF, we used 100 layers and survival rate $\alpha = 90\%$, which was optimal. For IS the computation time is controlled by the total number of particles. For SS-APF and APF it is controlled by number of particles per layer. For SS the running time is controlled via setting of M between 3 and 6. Reliability is the percentage of experiments that localized the box successfully, i.e. had at least one particle within 1mm and 1° of the true pose.

IS was unable to localize the box even after several minutes of computation. SS and SS-APF were able to localize the box within several seconds (0.3s and 5s respectively), with SS being approximately 15 times faster than SS-APF. APF localized the box within 100 seconds, which is approximately 300 times slower than SS and 20 times slower than SS-APF. These comparisons underline the impact of the Even_Density_Cover step (SS vs. SS-APF) and annealing schedule methodology (SS-APF vs. APF).

3) Multi-mode estimation: In this set of experiments we used data sets with 3 measurements from three adjacent sides of the box. Such data sets do not fully constrain the problem and the resulting posterior has four modes. We evaluated reliability of SS, SS-APF, APF, and IS (Fig. 11(a)). An experiment was considered successful if the approximation had at least one particle within 1mm and 1° of each of the four modes. For SS, the running time was varied by setting $M = 5$ to 7. Again, IS was unable to find all the modes even after several minutes. SS and SS-APF both were able to find all the modes, with SS-APF taking 20s, and SS being approximately 15 times faster (1.5s). APF was not completely reliable even after 10 minutes of computation, but it did reach reliability of 88%. APF was approximately 100 times slower than SS-APF and 1500 times slower than SS. We suspect the difference in performance would be even greater with more modes or whenever multiple modes need to be tracked over time.

4) Neighborhood size and shape: We evaluated the effect of δ_* and hyper-ellipse shape on the performance of Scaling Series. The hyper-ellipse shape is controlled by the position radius, r_{pos}, and the orientation radius, r_{ori}. We kept $r_{\text{pos}} = \delta_*$ in all experiments. Fig. 11(b) shows the effect of changing r_{ori}. The value computed via the Lipschitz constant (from Appendix A) was 0.5° (bright red line), with performance close to optimal. Optimal performance was achieved with $r_{\text{ori}} = 1°$ (bright blue line). This is likely due to the fact that measurements tend to land in the interior of box faces, hence the effective radius for Lipschitz constant computations is smaller than the actual box radius.

Fig. 12 shows the impact of δ_* and M on accuracy. In the left plot, each curve keeps δ_* constant, and varies running time by changing M. The performance with the predicted $\delta_* = 0.7\text{mm}$ is optimal and remains optimal with δ_* in the 0.7 to 1.5mm range.

In the right plot, each curve keeps M constant and varies δ_*. $M = 6$ was optimal converging to the minimum average

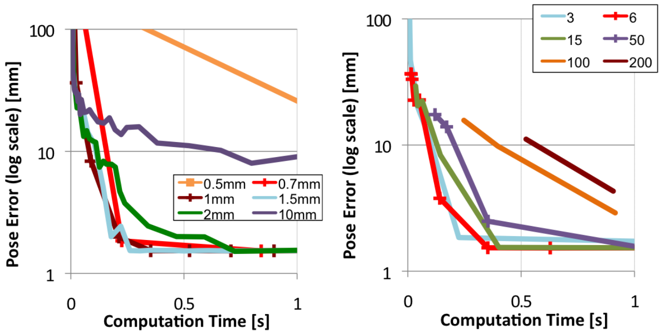
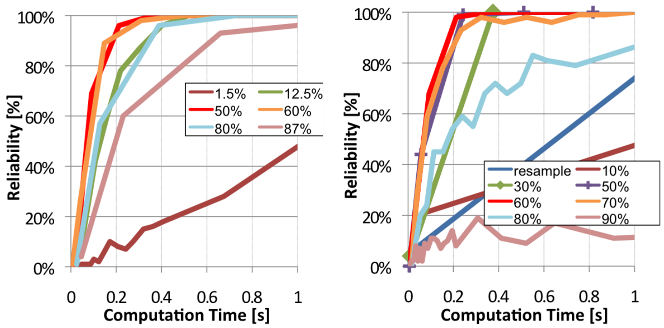


Fig. 12. Impact of δ_* (left) and M (right) on accuracy of SS.



(a) zoom factor

(b) pruning strategies

Fig. 13. (a) Effect of *zoom* factor on reliability. Legend shows $Vol(V_n)/Vol(V_{n-1})$ ratio. (b) Reliability for two different pruning strategies: resampling and thresholding. For thresholding, the legend shows the threshold ξ.

error of 1.5mm with $\delta_* = 1\text{mm}$ in 0.3s.

5) Zoom factor: The plot in Fig. 13(a) shows reliability vs. time for varying settings of $zoom$. The results are reported in terms of $Vol(V_n)/Vol(V_{n-1})$ ratio, which is easier to understand than $zoom$ itself. Ratio of 50% was optimal, which corresponds to $zoom$ setting shown in Alg. 1. However, ratios of 12.5% to 80% worked well.

6) Pruning: We compared the performance of Scaling Series with resampling and thresholding pruning strategies (Fig. 13(b)). For thresholding the legend shows values of ξ. Although not visible in the figure, different settings of M result in the same running time for the two different strategies. SS with resampling needs somewhere between $M = 2$ and $M = 3$ particles per δ-neighborhood. It is possible to extend Even_Density_Cover to work with non-integer values of M, which would allow for better performance with resampling pruning strategy. SS with thresholding on $\xi = 30\%$ to 70% needs $M = 3$ to 14 respectively. The optimal threshold was $\xi = 60\%$ (with $M = 6$) as predicted in Sect. IV-D2.

VI. Conclusions

We have considered the problem of global object localization via touch. Bayesian posterior estimation for objects in 6DOF has been known to be computationally expensive for this problem [6]. We have proposed an efficient approach, termed Scaling Series, that approximates the posterior by particles. It performs the estimation by successively refining the high probability region and scaling granularity of estimation from low to high. Our approach does not utilize any special properties of the manipulated objects and can be easily applied to any object represented as a polygonal mesh. We have demonstrated its portability by applying it to five different everyday objects on two robotic platforms.

For fully-constraining data sets, our approach performs the estimation in real time (under 1s) with very high reliability ($\geq 99\%$). At early stages of exploration, when the data set does not fully constrain the object, the resulting posterior is multi-modal. Running time in these cases depends on the precision desired and the size of the high probability region. However, our approach allows us to trade off precision of estimation for running time. Coarse estimates can be obtained quickly. As additional measurements arrive, the ambiguities are resolved and so more precise estimates can be obtained in a timely fashion.

We have provided analysis of convergence of the proposed algorithm along with strategies for parameter selection. We have also compared Scaling Series to a number of prior approaches. The results show that the proposed method outperforms prior art and is much more stable in multi-modal cases.

Similarly to [3], we expect that our approach can be extended to perform object identification from a set of known objects. Also, due to its stable performance with multi-modal posteriors that arise during exploration, we expect our approach to be particularly well suited for active exploration strategies that derive the optimal next sensing action based on prior data as in [7], [8].

The Scaling Series algorithm can be used with other applications and sensors. For example, in [57] we used Scaling Series for mobile manipulation during building navigation based on 2D laser range finders. In [58] we used Scaling Series for vehicle tracking based on 3D range data. In both cases, Scaling Series provided a significant improvement over state-of-the-art inference methods. In both of these works we included additional parameters in the Scaling Series filter. In [57] we used an articulated model of a door and estimated its opening angle along with robot's position. In [58], we estimated the number of moving vehicles in a previously unknown environment, as well as estimated vehicles' shape, position, and velocity. Similar techniques can be applied to touch based object localization when less information about the object shape is available a priori or when working with articulated objects.

Although in this paper we focused on the sense of touch exclusively, the presented approach can be naturally combined with other sensing modalities. For example, if a prior pose estimate is available from a vision system, it can be used to initialize samples of Scaling Series. If several sensing modalities are to be used simultaneously, one can perform sensor updates for each sensor within the same Scaling Series filter.

A number of aspects of the presented approach can be improved upon in future work. The running time of the algorithm depends linearly on the complexity of objects (i.e. number of faces in the mesh model). However, it is possible to implement efficiency improvements that only consider a small subset of faces during each measurement evaluation. So far experiments with moving objects have only been carried out in simulation, and so this aspect warrants further attention, although better hardware is likely to be required. Additional considerations will be needed if the object to be localized is placed into a cluttered environment, where the correspondence problem of measurements to objects will need to be solved. More work can go into devising better sensing procedures in order to reduce sensing time. In particular, it is possible to use compliant motions during exploration to reduce the time the robot has to travel to and from the object. However, more sophisticated sensor configuration will be required to make sure the robot does not contact the object with non-sensing surfaces during exploration.

Appendix A
Lipschitz Constants for Tactile Manipulation

In this section we provide bounds on Lipschitz constants of π and u. These bounds are used as explained in Sect. IV-C4 to set the shape of δ-neighborhoods. Theorem A1 relates the Lipschitz constants of π and u. Theorem A2 computes bounds on partial Lipschitz constants of u. For proof of Theorem A2 see [59].

Theorem A1: *If λ_π and λ_u are Lipschitz constants of π and u respectively, then $\lambda_\pi \leq \lambda_u/\sqrt{e}$. The same relationship holds for partial Lipschitz constants of π and u.*

Proof: We have $\pi(X) = \exp(-u^2(X)/2)$. Hence π can be expressed as a composition: $\pi = g \circ u$, where $g(u) := \exp(-u^2/2)$. Using chain rule, all partial derivatives of π can be written as $\frac{\partial \pi}{\partial x_i} = g'(u(X))\frac{\partial u}{\partial x_i}$. The (partial) Lipschitz constant can be computed as maximum (partial) derivative. For $g(u)$ we have $\lambda_g = \sup_u |g'(u)|$. As one can easily compute, this works out to $1/\sqrt{e}$. Thus using chain rule we obtain the desired result. □

Theorem A2: *The partial Lipschitz constants of $u(X)$ are bounded by:*

$$\begin{aligned} \lambda_{u,x}, \lambda_{u,y}, \lambda_{u,z} &\leq \sqrt{K}\frac{1}{\sigma_{\text{pos}}} \\ \lambda_{u,\alpha}, \lambda_{u,\beta}, \lambda_{u,\gamma} &\leq \sqrt{K}\sqrt{\frac{\mathcal{R}^2(\mathcal{O})}{\sigma^2_{\text{pos}}} + \frac{1}{\sigma^2_{\text{nor}}}}. \end{aligned} \tag{17}$$

Acknowledgments

We are grateful to Jaeheung Park, Irena Paschenko, Dongjun Shin, and Peter Thaulad for their help with the PUMA robot and to Peter Varvak, Torsten Kroeger, Alan Asbeck, and Morgan Quigley for their help with the most recent set of experiments. We give warm thanks to Jimmy Zhang, David Li, Jamie Schulte, and Francois Conti for help with the STAIR robot. Our sincere thanks to Sebastian Thrun, Andrew Y. Ng, David Stavens, Daphne Koller, Luis Sentis, and Vincent De Sapio for their feedback on earlier versions of this paper. We also thank anonymous reviewers for their insightful comments and one ICRA2006 anonymous reviewer in particular for proposing a simpler version of the Even Density Cover algorithm presented in this version of the paper. This work was supported by the Honda Motor Company.

References

[1] D. Kragic and H. I. Christensen, "Survey on visual servoing for manipulation," Royal Institute of Technology (KTH), Tech. Rep. ISRN KTH/NA/P–02/01–SE, 2002.

[2] J. R. Flanagan, M. C. Bowman, and R. S. Johansson, "Control strategies in object manipulation tasks," *Current Opinion in Neurobiology*, 2006.

[3] W. E. L. Grimson and T. Lozano-Perez, "Model-based recognition and localization from sparse range or tactile data," *Journal of Robotics Research*, 1983.

[4] O. Faugeras and M. Hebert, "A 3-d recognition and positioning algorithm using geometrical matching between primitive surfaces." in *Eighth Intl. Joint Conf. on Artificial Intelligence, Los Altos, CA*, 1983, pp. 996–1002.

[5] S. Shekhar, O. Khatib, and M. Shimojo, "Sensor fusion and object localization," in *Proc. of ICRA*, 1986.

[6] K. Gadeyne and H. Bruyninckx, "Markov techniques for object localization with force-controlled robots," in *ICAR*, 2001.

[7] S. Chhatpar and M. Branicky, "Particle filtering for localization in robotic assemblies with position uncertainty," in *IROS*, 2005.

[8] K. Hsiao, L. Kaelbling, and T. Lozano-Perez, "Task-driven tactile exploration," in *Proceedings of Robotics: Science and Systems*, Zaragoza, Spain, June 2010.

[9] A. Petrovskaya, O. Khatib, S. Thrun, and A. Y. Ng, "Bayesian estimation for autonomous object manipulation based on tactile sensors," in *IEEE International Conference on Robotics and Automation (ICRA)*, Orlando, Florida, USA, May 2006, pp. 707–714.

[10] ——, "Touch based perception for object manipulation," in *Robotics: Science and Systems (RSS), Robot Manipulation Workshop*, Atlanta, GA, USA, June 2007.

[11] C. Kemp, A. Edsinger, and E. Torres-Jara, "Challenges for robot manipulation in human environments [grand challenges of robotics]," *IEEE Robotics & Automation Magazine*, vol. 14, no. 1, pp. 20–29, 2007.

[12] A. Jain and C. Kemp, "Behavior-based door opening with equilibrium point control," in *RSS Workshop: Mobile Manipulation in Human Environments*, 2009.

[13] M. Prats, P. Sanz, and A. del Pobil, "Reliable non-prehensile door opening through the combination of vision, tactile and force feedback," *Autonomous Robots*, vol. 29, no. 2, pp. 1–18, August 2010.

[14] P. C. Gaston and T. Lozano-Perez, "Tactile recognition and localization using object models," MIT, AIM-705, Tech. Rep., 1983.

[15] H. Yau and C. Menq, "An automated dimensional inspection environment for manufactured parts using coordinate measuring machines," *International journal of production research*, vol. 30, no. 7, pp. 1517–1536, 1992.

[16] K. Gunnarsson and F. Prinz, "CAD model-based localization of parts in manufacturing," *Computer;(United States)*, vol. 20, no. 8, 1987.

[17] K. Gunnarsson, "Optimal part localization by data base matching with sparse and dense data," Ph.D. dissertation, Dept. of Mechanical Engineering, Carnegie Mellon Univ., May 1987.

[18] H. Pahk and W. Ahn, "Precision inspection system for aircraft parts having very thin features based on CAD/CAI integration," *The International Journal of Advanced Manufacturing Technology*, vol. 12, no. 6, pp. 442–449, 1996.

[19] M. Cho and T. Seo, "Inspection planning strategy for the on-machine measurement process based on CAD/CAM/CAI integration," *The International Journal of Advanced Manufacturing Technology*, vol. 19, no. 8, pp. 607–617, 2002.

[20] Z. Xiong, "Workpiece Localization and Computer Aided Setup System," Ph.D. dissertation, The Hong Kong University of Science and Technology, 2002.

[21] J. Hong and X. Tan, "Method and apparatus for determining position and orientation of mechanical objects," May 4 1993, uS Patent 5,208,763.

[22] B. Horn, "Closed-form solution of absolute orientation using unit quaternions," *JOSA A*, vol. 4, no. 4, pp. 629–642, 1987.

[23] C. Menq, H. Yau, and G. Lai, "Automated precision measurement of surface profile in CAD-directed inspection," *Robotics and Automation, IEEE Transactions on*, vol. 8, no. 2, pp. 268–278, 1992.

[24] Y. Chu, "Workpiece localization: Theory, algorithms and implementation," Ph.D. dissertation, The Hong Kong University of Science and Technology, 1999.

[25] Z. Xiong, M. Wang, and Z. Li, "A near-optimal probing strategy for workpiece localization," *Robotics, IEEE Transactions on*, vol. 20, no. 4, pp. 668–676, 2004.

[26] Y. Huang and X. Qian, "An Efficient Sensing Localization Algorithm for Free-Form Surface Digitization," *Journal of Computing and Information Science in Engineering*, vol. 8, p. 021008, 2008.

[27] L. Zhu, H. Luo, and H. Ding, "Optimal Design of Measurement Point Layout for Workpiece Localization," *Journal of Manufacturing Science and Engineering*, vol. 131, p. 011006, 2009.

[28] L. Zhu, Z. Xiong, H. Ding, and Y. Xiong, "A distance function based approach for localization and profile error evaluation of complex surface," *Journal of manufacturing Science and Engineering*, vol. 126, p. 542, 2004.

[29] Y. Sun, J. Xu, D. Guo, and Z. Jia, "A unified localization approach for machining allowance optimization of complex curved surfaces," *Precision Engineering*, vol. 33, no. 4, pp. 516–523, 2009.

[30] C. Corcoran and R. Platt, "A measurement model for tracking handobject state during dexterous manipulation," in *ICRA*, 2010.

[31] R. Platt Jr, F. Permenter, and J. Pfeiffer, "Inferring hand-object configuration directly from tactile data," in *Mobile Manipulation Workshop at ICRA*, 2010.

[32] M. Charlebois, K. Gupta, and S. Payandeh, "Curvature based shape estimation using tactile sensing," in *IEEE International Conference on Robotics and Automation*, 1996, pp. 3502–3507.

[33] A. Bicchi, A. Marigo, and D. Prattichizzo, "Dexterity through rolling: Manipulation of unknown objects," in *ICRA*, 1999, pp. 1583–1588.

[34] M. Kaneko and T. Tsuji, "Pulling motion based tactile sensing," in *Algorithmic and computational robotics: new directions: the fourth Workshop on the Algorithmic Foundations of Robotics*. AK Peters, Ltd., 2001, p. 157.

[35] M. Moll, "Shape Reconstruction Using Active Tactile Sensors," Ph.D. dissertation, University of California, 2002.

[36] P. Slaets, J. Rutgeerts, K. Gadeyne, T. Lefebvre, H. Bruyninckx, and J. De Schutter, "Construction of a geometric 3-D model from sensor measurements collected during compliant motion," in *Proc. of ISER*, 2004.

[37] M. Schaeffer and A. M. Okamura, "Methods for intelligent localization and mapping during haptic exploration," in *Proc. of the IEEE International Conference on Systems, Man and Cybernetics*, 2003.

[38] A. Doucet and N. De Freitas, *Sequential Monte Carlo Methods in Practice*. Springer, 2001.

[39] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. MIT Press, 2005.

[40] M. Montemerlo, "Fastslam: A factored solution to the simultaneous localization and mapping problem with unknown data association," Ph.D. dissertation, Robotics Institute, Carnegie Mellon University, 2003.

[41] V. Lepetit and P. Fua, "Monocular model-based 3D tracking of rigid objects," *Foundations and Trends® in Computer Graphics and Vision*, vol. 1, no. 1, pp. 1–89, 2005.

[42] C. Harris, "Tracking with Rigid Objects," *Active Vision*, pp. 59–73, 1993.

[43] T. Drummond and R. Cipolla, "Real-time visual tracking of complex structures," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 7, pp. 932–946, 2002.

[44] M. Fischler and R. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[45] A. Rahimi and T. Darrell, "Location Estimation with a Differential Update Network," in *NIPS*, 2002, pp. 1049–1056.

[46] A. Davison, "Real-Time Simultaneous Localisation and Mapping with a Single Camera," in *ICCV*, 2003.

[47] M. Isard and A. Blake, "A smoothing filter for condensation," *Computer Vision—ECCV'98*, pp. 767–781, 1998.

[48] J. Deutscher, A. Blake, and I. Reid, "Articulated body motion capture by annealed particle filtering," in *CVPR*, 2000.

[49] J. Deutscher and I. Reid, "Articulated body motion capture by stochastic search," *International Journal of Computer Vision*, vol. 61, no. 2, pp. 185–205, 2005.

[50] A. Balan, L. Sigal, and M. Black, "A quantitative evaluation of video-based 3D person tracking," in *ICCCN*, vol. 5, 2005, pp. 349–356.

[51] D. J. C. MacKay, "Introduction to Monte Carlo methods," in *Learning in Graphical Models*, ser. NATO Science Series, M. I. Jordan, Ed. Kluwer Academic Press, 1998, pp. 175–204.

[52] C. Olson, "Probabilistic self-localization for mobile robots," *IEEE Transactions on Robotics and Automation*, vol. 16, no. 1, pp. 55–66, 2000.

[53] S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for on-line non-linear/non-gaussian bayesian tracking," *IEEE Transactions on Signal Processing*, 2002.

[54] J. Geweke, "Bayesian inference in econometric models using Monte Carlo integration," *Econometrica: Journal of the Econometric Society*, pp. 1317–1339, 1989.

[55] N. Gordon, "Bayesian methods for tracking," Ph.D. dissertation, University of London, 1993.

[56] A. Petrovskaya, *Robotic Perception via Contact*. http://cs.stanford.edu/people/petrovsk/tactile.html, 2011.

[57] A. Petrovskaya and A. Ng, "Probabilistic mobile manipulation in dynamic environments, with application to opening doors," in *International Joint Conference on Artificial Intelligence (IJCAI)*, Hyderabad, India, January 2007.

[58] A. Petrovskaya and S. Thrun, "Model based vehicle detection and tracking for autonomous urban driving," *Autonomous Robots*, vol. 26, no. 2, pp. 123–139, April 2009. [Online]. Available: http://dx.doi.org/10.1007/s10514-009-9115-1

[59] A. Petrovskaya, "Towards dependable robotic perception," Ph.D. dissertation, Stanford University, Stanford, CA, June 2011.

Anna Petrovskaya received her Doctorate degree in Computer Science from Stanford University, Stanford, CA, USA, in 2011. Her research focuses on model based Bayesian perception for robotic applications. She has developed new efficient algorithms for tactile object localization, mobile manipulation and vehicle tracking. Her contributions to Robotics have been recognized by the Stanley Scholar fellowship and the Achievement Rewards for College Scholars fellowship.

Oussama Khatib received his Doctorate degree in Electrical Engineering from SupAero, Toulouse, France, in 1980. He is Professor of Computer Science at Stanford University. His work on advanced robotics focuses on methodologies and technologies in human-centered robotics including humanoid control architectures, human motion synthesis, interactive dynamic simulation, haptics, and human-friendly robot design. He is Co-Editor of the Springer Tracts in Advanced Robotics series, and has served on the Editorial Boards of several journals as well as the Chair or Co-Chair of numerous international conferences. He coedited the Springer Handbook of Robotics, which received the PROSE Award for Excellence in Physical Sciences & Mathematics and was also the winner in the category Engineering & Technology. He is a Fellow of IEEE and has served RAS as a Distinguished Lecturer, as a member of the Administrative Committee, and as the Program Chair of ICRA 2000. He is the President of the International Foundation of Robotics Research (IFRR) and a recipient of the Japan Robot Association (JARA) Award in Research and Development. Professor Khatib received the 2010 IEEE RAS Pioneer Award in Robotics and Automation for his fundamental pioneering contributions in robotics research, visionary leadership, and life-long commitment to the field.