Subjective Localization with Action Respecting Embedding

Michael Bowling[1], Dana Wilkinson[2], Ali Ghodsi[3], and Adam Milstein[2]

[1] Department of Computing Science, University of Alberta
bowling@cs.ualberta.ca
[2] School of Computer Science, University of Waterloo
{d3wilkin,ahpmilst}@uwaterloo.ca
[3] Department of Statistics and Actuarial Science, University of Waterloo
aghodsib@uwaterloo.ca

Summary. Robot localization is the problem of how to estimate a robot's pose within an objective frame of reference. Traditional localization requires knowledge of two key conditional probabilities: the motion and sensor models. These models depend critically on the specific robot as well as its environment. Building these models can be time-consuming, manually intensive, and can require expert intuitions. However, the models are necessary for the robot to relate its own subjective view of sensors and motors to the robot's objective pose. In this paper we seek to remove the need for human provided models. We introduce a technique for *subjective localization*, relaxing the requirement that the robot localize within a global frame of reference. Using an algorithm for action-respecting non-linear dimensionality reduction, we learn a subjective representation of pose from a stream of actions and sensations. We then extract from the data natural motion and sensor models defined for this new representation. Monte Carlo localization is used to track this representation of the robot's pose while executing new actions and receiving new sensor readings. We evaluate the technique in a synthetic image manipulation domain and with a mobile robot using vision and laser sensors.

1 Introduction

A key problem in mobile robotics is localization: estimating a robot's pose while it moves and senses in the world. Knowledge of a robot's position in its environment is one of the most basic requirements for many autonomous tasks. The majority of localization techniques focus on *objective localization*, where the pose is estimated in terms of a human defined global frame of reference. For example, pose may be defined as the position and orientation on a two-dimensional Cartesian map with units in meters. In this paper, we seek to relax this notion of localization.

One of the most successful approaches to objective localization uses probabilities to model all aspects of a robot's uncertainty, including the current pose estimate, the effect of actions, and the information provided by sensors. Rules of probabilistic inference can then be applied in a straightforward fashion to maintain an estimate of the robot's location. Approaches of this type often restrict the form of the models (e.g., Gaussian distributions in Kalman filters [Kal60]) or use various approximation techniques (e.g., sampling in Monte Carlo localization [FBDT99]), to allow inference, and thus localization, to be computationally feasible.

A key prerequisite for all probabilistic approaches are models of the uncertainty in the robot's motion and sensors. Classical kinematics defines the expected global motion of the robot when a particular control is applied to it. But kinematics requires many assumptions in its deterministic calculations (e.g., infinite friction) that do not hold in practice. Hence, robot motion is uncertain. Likewise, there are many uncontrollable and unpredictable factors (e.g., acoustic reflectance of a surface with sonar, or ambient lighting with vision) that effect readings from sensors. Hence, robot sensing is also uncertain. Probabilistic models of these uncertainties form the basis for inference (which drives the localization). Unfortunately, these models are often not easy to build. They can require extensive knowledge of the robot's kinematics or sensors, which may not be known or easily described. They may require time-consuming manual measurements to estimate characteristics of noise or to build a map of sensor readings over the environment. Finally, by definition, a well constructed model must be specific to the particular hardware used. Modifying the robot platform invalidates these laboriously constructed models and new models must be created. For example, changing from a wheeled robot to a legged robot obviously invalidates the motion model. Changing from a sonar to a laser, or from a laser to a camera will require replacement of the sensor model. Even minor changes, such as inflating the tires on the robot, or replacing its camera with one of a different model, will require expert modifications to the various models. Recent work has examined techniques for automatically calibrating some of these models (e.g., [RT99], [MTKW02], [EP04], [SS05]), but no current method exits to calibrate these models for objective localization without considerable expert knowledge.[4]

This paper examines the problem of *subjective localization*. We relax the requirement that the robot must estimate its pose in terms of a global frame of reference. Instead, the choice of representation is left as part of the localization problem. This relaxation allows the robot to learn both motion and sensor models as the models can be defined purely in terms of its own subjective motor and sensor values. Although objective localization may be necessary for certain tasks, not all tasks require knowledge of an objective position. Delivery tasks, for example, need only recognize location with respect to locations visited in the past. A robot can be given a guided tour of its environment ("getting its bearings") and informed of salient locations along the tour which can then be labeled in its subjective map.

The problem of subjective localization will be tackled with a four-step process. We first gather data of the robot moving and sensing in its world. We then use this data to learn both an appropriate frame of reference for localization as well as the actual trajectory the robot followed during the data gathering. We then learn motion and

[4] Special mention should be made of the work of Stronger and Stone [SS05], which learns motion and sensor models starting with only an inaccurate motion model. Their approach is still quite knowledge intensive, using a human-defined preprocessing step to simplify the complex image sensor down to a single estimate of distance to a beacon.

sensor models in this frame of reference from the training data and the learned trajectory. Finally, we incorporate these models into Monte Carlo Localization (MCL), a probabilistic localization technique. The cornerstone of this approach is extracting a subjective frame of reference from a trace of sensorimotor data. This is solved with Action Respecting Embedding (ARE) [BGW05], a technique for non-linear dimensionality reduction which finds low-dimensional descriptions of the robot's pose in a frame where actions correspond to simple transformations.

The rest of this paper is organized as follows. Section 2 provides an overview of Monte Carlo localization. Section 3 summarizes the Action Respecting Embedding algorithm, which extracts the subjective representation. Section 4 describes the learning of motion and sensor models in this new frame of reference, which can then be used in MCL. Section 5 demonstrates the effectiveness of this approach, both in a synthetic image manipulation domain and with a mobile robot using first a camera and then a laser as the primary sensor. Section 6 concludes.

2 Monte Carlo Localization

Monte Carlo Localization (MCL) [FBDT99] is a method for estimating the posterior distribution of the robot's pose conditioned on the robot's actions and sensor readings. It relies on the Markovian assumption that the past and the future are conditionally independent given the present. MCL is an implementation of a recursive Bayes filter. If x_t is the location at time t, z_t is the sensor data at time t, and u_t is the motion data at time t then the posterior distribution becomes:

$$\mathrm{Bel}(x_t) = p(x_t|z_T, u_T) \tag{1}$$

where $z_T = z_1, \ldots, z_t$ and, similarly, $u_T = u_1, \ldots, u_{t-1}$. For objective localization the sensor data is usually in the form of range data, such as laser range-finder readings, however any type of sensor for which the proper kind of model exists is admissible. The motion data is usually the report from the robot's odometers, but again, any data with an appropriate model will satisfy the equation.

For a recursive Bayes' filter, a recursive formula is necessary, so Equation 1 is converted, using a combination of Bayes' rule and the Markovian assumption, into:

$$Bel(x_t) = (1/Z)\, p(z_t|x_t) \int p(x_t|x_{t-1}, u_t) Bel(x_{t-1})\, dx_{t-1}, \tag{2}$$

where Z is a normalization term. $p(x_t|u_t, x_{t-1})$ is called the motion model, the probability of a resulting pose given a starting pose and an action. $p(z_t|x_t)$ is called the sensor model, the probability of receiving a particular sensor reading given the robot's pose. If these two models exist then MCL can be performed.

Unfortunately, virtually all robots operate in a continuous space, so the integral in Equation 2 is impossible to compute directly. In order to solve the problem, MCL approximates the continuous space with a finite set of samples or "particles". At each time-step the set of samples is moved probabilistically according to the motion model. The samples are then annotated with a weight determined by the sensor model. The weight of each sample is the probability of receiving the observed sensor reading given that the robot is at the location represented by the particle. Finally, the particles are resampled according to their weight. Resampling generates the new set of samples by choosing a particle with probability proportional to its weight with replacement. Although MCL is obviously only correct as the number of samples approaches infinity, it is often accurate for a relatively small number of samples.

MCL is a common technique for objective localization, where the motion model and sensor model are constructed by hand or through experimentation. It can be used equally well for subjective localization if one has appropriate motion and sensor models in a subjective frame of reference. Section 3 deals with finding such a frame of reference, while Section 4 details the learning of the required models.

3 Action Respecting Embedding

High-dimensional data sets, such as a sequence of images or scans from a laser range-finder, can usually be characterized by a low-dimensional representation that is related to the process generating the data. For example, one low-dimensional representation for image data might correspond to the degrees of freedom of the platform moving the camera which gathered the data. Such a low-dimensional representation of the sensor readings might be an ideal frame of reference for subjective localization. The goal, then, is to take a temporal sequence of sensor readings $z_1, \ldots, z_n$ with associated control actions, $u_2, \ldots, u_n$, and find a low-dimensional representation for $z_1, \ldots, z_n$ that would be appropriate for localization.

Recently, *non-linear manifold-learning* techniques have been used to map high-dimensional datasets, such as sensor readings, into smaller dimensional spaces. Semidefinite Embedding (SDE) [WS04] is one such technique. SDE learns a kernel matrix, which represents a non-linear projection of the input data into a more linear representation. It then uses Kernel PCA [SS02], a generalization of principle components analysis to feature spaces represented by kernels, to extract out a low-dimensional representation of the data. The kernel matrix, K, is learned in SDE by solving a semidefinite program with a simple set of constraints. The most important constraints encode the common requirement in dimensionality reduction that the non-linear embedding should preserve local distances. In other words, nearby points in the original input space should remain nearby in the resulting feature representation. Therefore, SDE requires a distance metric $||\cdot||$ on the original input space, and uses this metric to construct a k-nearest neighbors graph. It then adds constraints into the semidefinite program to ensure that the distance between neighbors is preserved. The optimization maximizes the trace of K, i.e., the variance of the learned feature representation, which should minimize its dimensionality.

SDE, though, ignores two important pieces of knowledge about our data: the temporal ordering of the input vectors z_i, and the action labels u_i. Therefore, SDE doesn't require temporally nearby input points to be spatially nearby in the feature representation. Also, SDE won't enforce the extracted space to be one where the robot's actions have a simple interpretation. The recent Action Respecting Embedding (ARE) algorithm uses the aforementioned knowledge to address these issues.

Formally, ARE takes a set of D-dimensional input vectors, $z_1, \ldots, z_n$ (i.e., sensor readings, in temporal order) along with associated discrete actions $u_1, \ldots, u_{n-1}$ (where action u_i was executed between input z_i and input z_{i+1}), and computes a set of d-dimensional output vectors $x_1, \ldots, x_n$ in one-to-one correspondence with the input vectors that provide a meaningful embedding in $d < D$ dimensions. ARE is similar to SDE but extends it in two key ways. First, it exploits the knowledge that the sensor readings are given in a temporal sequence by building an improved neighborhood graph based on each input's distances to its temporal neighbors using an arbitrary local distance metric[5]. Second, it constrains the embedding to respect the action labels that are associated with adjacent pairs of observations. This ensures that the actions have a simple interpretation in the resulting feature space.

This second enhancement of ARE is the critical feature for subjective localization. ARE constrains the learned manifold to be a space where the actions correspond to transformations consisting only of rotation and translation in that space[6]—in other words, every action is required to be a distance-preserving transformation for all inputs in the learned feature space. Letting $\Phi(z_i)$ denote input z_i's representation in this learned feature space, we require u's transformation, f_u, to satisfy:

$$\forall i, j \;\; ||f_u(\Phi(z_i)) - f_u(\Phi(z_j))|| = ||\Phi(z_i) - \Phi(z_j)||. \tag{3}$$

Now, let $u = u_i = u_j$, so $f_u(\Phi(z_i)) = \Phi(z_{i+1})$ and $f_u(\Phi(z_j)) = \Phi(z_{j+1})$. Hence, constraint 3 becomes:

$$||\Phi(z_{i+1}) - \Phi(z_{j+1})|| = ||\Phi(z_i) - \Phi(z_j)||. \tag{4}$$

In terms of the kernel matrix, this can be written as:

$$\begin{aligned} \forall i, j \quad u_i = u_j \Rightarrow\; & K_{(i+1)(i+1)} - 2K_{(i+1)(j+1)} + K_{(j+1)(j+1)} \\ =\; & K_{ii} - 2K_{ij} + K_{jj}. \end{aligned} \tag{5}$$

Add constraint 5 to the SDE optimization problem to get the ARE algorithm shown in Table 1.

Algorithm: ARE$(||\cdot||, (z_1, \ldots, z_n), (u_2, \ldots, u_n))$

Construct neighbor graph, N**, according to [BGW05].**

Maximize $\mathrm{Tr}(K)$ **subject to** $K \succeq 0$, $\sum_{ij} K_{ij} = 0$,

$$\begin{aligned} &\forall ij \;\; N_{ij} > 0 \vee [N^T N]_{ij} > 0 \Rightarrow \\ &\qquad K_{ii} - 2K_{ij} + K_{jj} \le ||z_i - z_j||^2 \text{, and} \\ &\forall ij \quad u_i = u_j \Rightarrow K_{(i+1)(i+1)} - 2K_{(i+1)(j+1)} + K_{(j+1)(j+1)} \\ &\qquad\qquad = K_{ii} - 2K_{ij} + K_{jj} \end{aligned}$$

Run Kernel PCA with learned kernel, K**.**

Table 1. Algorithm: Action Respecting Embedding (ARE).

[5] We have found that ARE is fairly robust to the choice of distance metrics, and use simple Euclidean distance for all of the experiments in this paper.

[6] Notice this is not requiring the actions in the objective space to be rotations and translations, since ARE is learning a non-linear feature representation.

4 Subjective Localization

Recall that the subjective localization problem involves both determining an appropriate subjective frame of reference for localization and then tracking the robot's position in that representation. In the work presented in this paper, ARE is used to learn the frame of reference. A completely unsupervised stream of data from a robot acting in the world (consisting of a stream of sensor readings, $z_1, \ldots, z_n$, and associated actions, $u_1, \ldots, u_{n-1}$, which are elements of some set of discrete actions) will be used as input to ARE in order to learn an appropriate subjective representation.

In order to perform localization with this representation, a motion model and sensor model must be computed. ARE, though, provides more than just a coordinate system. It also provides the actual d-dimensional embedded points, $x_1, \ldots, x_n$, that correspond to the trajectory the robot followed in the data-gathering phase. This trajectory—along with the robot's sensations, $z_1, \ldots, z_n$, and actions, $u_1, \ldots, u_n$—can be used to learn the models from the training data. Both models will be learned in a similar fashion. We will first estimate the expectation of the model and then use the error to estimate a noise component. We begin with the motion model.

4.1 Motion Model

The motion model is the posterior distribution $p(x_t|u_t, x_{t-1})$. Since this model will be used in a particle filter, it is only necessary to be able to draw a sample, $\hat{x}$, from the model, given a u_t and x_{t-1}, i.e.:

$$\hat{x} \sim p(x_t|u_t, x_{t-1}).$$

First, separate the model into an expectation plus a noise component.

$$\hat{x} \sim E(x_t|u_t, x_{t-1}) + \eta(x_t|u_t, x_{t-1})$$

Now make the simplifying assumption that the noise depends only on the action and not on the previous pose. This gives the form:

$$\hat{x} \sim E(x_t|u_t, x_{t-1}) + \eta(x_t|u_t) \tag{6}$$

We can now learn the model by learning the expectation component, then using the sample errors to estimate the noise component.

Consider some action u. Every t where $u_t = u$ gives one sample, x_t and x_{t-1}, from the distribution $p(x_t|u, x_{t-1})$. Using these sample points, a function of x_{t-1} is desired that gives a close estimate of x_t. ARE explicitly includes constraints that ensure such a function exists and is a simple rotation plus a translation in the learned representation. We can recover these functions by solving an optimization problem to find the corresponding rotation matrix A_u and translation vector b_u such that $f_u(x) = A_u x + b_u$. Formally,

$$\textbf{Min} \sum_{t:u_t=u} ||A_u x_{t-1} + b_u - x_t||^2 \ \textbf{s.t.}\ A_u^T A_u = I \text{ (i.e., } A \text{ is a rotation)}$$

This problem is similar to the extended orthonormal Procrustes problem [SC70] and has a closed form solution. Let X_u be a matrix whose columns are x_{t-1} for all t such that $u_t = u$, and let Y_u be a matrix whose columns are x_t for the same t. The following is the solution to this optimization problem:

$$A_u = VW^T \quad \text{where} \quad VSW^T = \text{svd}\left(Y_u^T \left(I - \frac{ee^T}{d}\right) X_u\right) \tag{7}$$

$$b_u = (Y_u - A_u X_u)^T e/d, \tag{8}$$

where $\text{svd}(\cdot)$ is the singular value decomposition and e is a column vector with d ones. Now the expected motion can be defined as:

$$E(x_t|u_t, x_{t-1}) = A_{u_t} x_{t-1} + b_{u_t}. \tag{9}$$

Since we only included the top d principal components of the output of ARE, this model of the expected motion won't be exact. The errors in the learned transformation can be used to build a model of the motion noise. Consider again some action u, let ξ_t be the residual error for action u on x_t:

$$\xi_t = A_u x_{t-1} + b_u - x_t \qquad \text{where} \qquad \sum_{t:u_t=u} \xi_t = \mathbf{0}.$$

The motion noise can be modeled as a zero-mean multivariate Gaussian, where the covariance matrix can be estimated directly from the samples ξ_t. Formally:

$$\eta(x_t|u_t) \sim N(\mathbf{0}, \Sigma_{u_t}), \tag{10}$$

where:

$$\Sigma_{u_t}(i,j) = \sum_{t:u_t=u} \xi_t(i)\xi_t(j).$$

Combining Equations 6, 9, and 10 gives the complete motion model.

4.2 Sensor Model

The sensor model is the probability distribution $p(z_t|x_t)$. In the context of a particle filter, the density of the distribution at z_t must be provided for a given x_t. In estimating this model from the data a few assumptions must be made. Notice that ARE doesn't take the images directly as its input, but rather uses an image's distance to every other image as a kind of feature representation. We will use the same representation for new observations, computing a feature vector:

$$\bar{z}_t(i) = ||z_t - z_i|| \quad \forall i = 1 \dots n.$$

The best way to view this feature vector is that it provides a crude estimate of the "distance" of the robot's pose to the previous poses, $x_1, \dots, x_n$. The additional assumption is required that each of the components of the feature vector are independently distributed[7]. That is, each is an independent estimate of the "distance" to a past pose. The final assumption is that this probability only depends upon the *distance* to the specific past pose in the subjective representation[8], i.e., $||x_t - x_i||$. These assumptions combine to give the following form for the model. Let $d_{ti} = ||x_t - x_i||$:

[7] This assumption, while almost certainly incorrect, is similar to the common MCL assumption (often necessary for tractability) that sensor readings are independent.

[8] This is not an unreasonable assumption, since ARE explicitly constrains distances in the subjective representation $||x_i - x_j||$ by observed image distances $||z_i - z_j||$.

$$p(z_t|x_t) = p(\bar{z}_t|x_t) = \Pi_{i=1}^{n} p(\bar{z}_t(i)|x_t) = \Pi_{i=1}^{n} p\left(\bar{z}_t(i)|d_{ti}\right). \tag{11}$$

Now to estimate a Gaussian model for the conditional random variable $\bar{z}_t(i)|d_{ti}$. Consider again the training trajectory, each x_t gives one sample for this joint distribution: $\bar{z}_t(i)$ and d_{ti}. To build a Gaussian model, for each landmark, i, use regression to fit a low-degree polynomial determining the distribution mean as a function of distance ($\mu_i(d_{ti})$)[9]. Then take the mean of the squared errors to estimate distribution variance (σ_i^2). This gives the following Gaussian density function:

$$\bar{z}_t(i)|d_{ti} \sim N(\mu_i(d_{ti}), \sigma_i^2). \tag{12}$$

Combining Equations 11 and 12 gives the sensor model.

4.3 Using the Models

The final step of the technique is to use the motion and sensor models with Monte Carlo localization to track the robot's position in the learned subjective space. The only detail left to be addressed is the initial distribution for localization. Since we processed the data after a single training run, we know our exact position in the subjective representation, x_n. All the samples in MCL are initialized to this point.

In the end, the subjective localization procedure has three configurable parameters: the dimensionality of the subjective representation, d, the degree of polynomial used in the sensor model, and the number of particles used by MCL. Overall, the procedure has a small number of parameters and, as seen in the next section, can actually localize in a number of different situations with a variety of parameter settings.

5 Results

Here, the algorithm from Section 4 is applied to two different domains. The first is IMAGEBOT a synthetic image manipulation domain. The second is a mobile robot, demonstrating localization with first a camera, then a laser range-finder as the primary sensor. First the domains are described followed by the experiments with the results of localization. Then a measure of accuracy is presented that is appropriate for subjective localization, showing accuracy across a variety of experiments. Finally, we show the robustness of the algorithm to the choice of its few parameters.

5.1 The Domains

We explored subjective localization in two different domains.

Image based (IMAGEBOT).

Given an image, imagine a virtual robot that observes a small patch on that image and takes actions to move this patch around the larger image. This "image robot" provides an excellent domain in which subjective localization can be rigorously tested while having obvious corollaries to mobile robotics.

For these experiments, IMAGEBOT will always be viewing a 100 by 100 patch of a 2048 by 1536 image. All the experiments use the image from Figure 1. IMAGEBOT has four translation actions and two zoom actions. The allowed translations are forward, backward, left and right, each by 25 pixels. The zoom changes the scale of

[9] This is very similar to the sensor model construction by Stronger and Stone [SS05].

the underlying image by a factor of $2^{1/8}$ or $2^{-1/8}$. Since we are interested in noisy actions, zero-mean Gaussian noise is added to the magnitude of the change of any of the actions with a standard deviation of one-tenth of the mean change.

Fig. 1. IMAGEBOT's world.

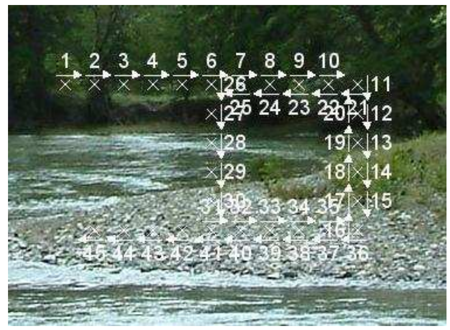

Fig. 2. A 45-action IMAGEBOT trajectory.

Mobile Robot.

Experiments were performed on an ActivMedia Pioneer 3 DX8 robot equipped with an ordinary web camera and a laser range-finder. A series of predefined actions were used to move the robot up and down a a corridor with data being collected after each action. Additionally, after each action was performed the robot's position was manually measured to discover actual error. We performed experiments using the camera as the only sensor, then using the laser as the only sensor.

5.2 Experiments

In all experiments, a dataset is gathered by executing a sequence of actions and receiving the associated sequence of sensor readings. After each action, measurement of objective location is taken—used later to compute a measure of accuracy. The sequence is split into two sets, training and test. The training set is used by ARE to extract a subjective representation and associated trajectory. Motion and sensor models are learned as described in Section 4. Finally, the models are used in MCL to localize given the test set. The mean of the particles after every given action and observation is used as the estimated position in the subjective frame of reference.

In order to extract a model of noise, the training data needs to contain examples of executing the same action from approximately the same location. Since the points after taking this action will be in various locations, the noise of the motion model can then be reconstructed. Therefore, each dataset begins by taking repeated short sequences of actions (such as going forward three steps then backward three steps), ensuring the training data includes a representation of noise in the robot's actions.

Image Based (IMAGEBOT).

In the IMAGEBOT domain three different paths were examined, each path was generated three times, each different due to noise. The first path was a simple line, where IMAGEBOT executed forward and backward actions. The second was an "A" shaped path using forward, backward, left, and right (an example of this trajectory is shown

Figure 2). The last was the same “A” shaped path where the right and left actions were replaced by zoom-in and zoom-out actions, respectively. In all cases, the test data involved retracing ImageBot’s steps back over its path. This involves different observations, though, as the actions are noisy.

Figure 3 shows an example “A” shaped path (the “A” is tilted to the right) in objective coordinates. The dotted line shows the training data with the trajectory starting in the upper left. The solid line shows the test data, a reversed “A” starting from the bottom left. Note that noise prevents the two paths from exactly lining up.

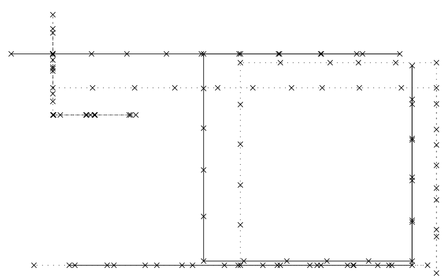

Fig. 3. ImageBot’s “A” shaped trajectory in objective coordinates. The dotted line is the training data, the solid line is the test data.

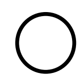

Fig. 4. Subjective localization on the “A” trajectory. Dotted line is ARE’s trajectory on training data, solid line is predicted location using MCL on test data.

Figure 4 shows the results of using the data from Figure 3 with our subjective localization technique. The dotted line shows the trajectory that resulted from running ARE on the training data in the learned frame of reference. The solid line is the predicted points from MCL while receiving images and actions from the test data. The circled cloud of points point shows the set of 100 particles in MCL at that point in the trajectory. The learned trajectory corresponds strongly with the objective trajectory, and the localized trajectory follows along appropriately. In the next section we investigate a quantitative measure of localization accuracy showing the results on this and the other trajectories.

Mobile Robot.

There was one simple path studied with the Pioneer robot, but two experiments were performed with it. In the first, observations were 160x120 pixel images from the camera. In the second, observations were the 180 distance estimates from the laser range-finder. Training and test paths were the same as the first path of ImageBot: a simple forward and backward trajectory. Figure 5 shows the consecutive images taken as the robot traversed this path. The top row (left to right) shows images as the robot moves forward down its path. The bottom row (right to left) shows the continuation of the trajectory as the robot moves back up the path.

Figure 6 shows ARE’s learned trajectory (dotted line) and the predicted trajectory from localization on the test set (solid line) using the camera as sensor input. The objective space corresponds to a single primary dimension, and the trajectories

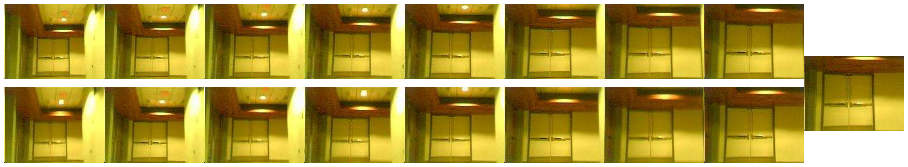

Fig. 5. Images gathered from the robot moving forward (top) then backward (bottom).

correctly capture this as the x dimension in the plot. The learned and predicted trajectories look qualitatively similar when the laser is used as sensor input and so are not shown here.

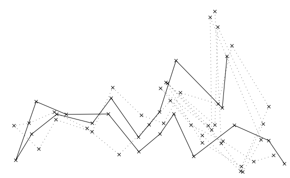

Fig. 6. Subjective localization of the robot using the camera. Dotted line is ARE's trajectory on training data, solid line is predicted location using MCL on test data.

5.3 Accuracy

Accuracy is a measure for evaluating localization performance. In objective localization, this amounts to comparing the predicted position to hand-measured objective positions and reporting the mean error. For subjective localization, this is not possible as the robot's location is only known in a subjective frame of reference. This makes it difficult to measure the accuracy of an algorithm. As mentioned in the introduction, one use for a subjective representation is for recognizing locations visited in the past. In particular, a new position in the subjective frame of reference can be compared to previous training points. Distance in the subjective space can be used to estimate which training point we expect to be closest to in objective space.

A method for evaluating subjective localization now becomes clear. For a given predicted subjective location, find the closest point in the training data to this location and consider this an objective prediction. The error is simply the distance in objective coordinates between the robot's true (measured) location and this prediction from the training set. This gives a measure of accuracy in objective terms (Note, it is generally impossible to achieve zero error). For comparison, an oracle score can also be computed. Look at the actual objective positions of each point in the test data and determine the closest training point, using this distance as the oracle error. Any measure of accuracy can be compared to this oracle's accuracy. As another baseline, compute the error of a random subjective localization algorithm that chooses a random training point as the prediction of its location. These two baselines, oracle and random, can be used to evaluate the accuracy of any subjective localization method.

Table 2 shows the accuracy results for all three paths in IMAGEBOT averaged over three datasets each with ten complete runs of MCL. Table 3 shows the accuracy results for both camera and laser-based robot experiments, averaged over ten complete runs of MCL. In IMAGEBOT, the accuracies are within approximately 10 pixels, a vast improvement over the random baseline and not far from the oracle's accuracy. With the mobile robot, the accuracy with the camera is approximately 150mm, an order of magnitude improvement over random and about an order of magnitude behind the oracle performance. The performance with the laser range-finder is not quite as strong, but still demonstrates effective localization.

	Mean Error (Pixels)		
	Oracle	ARE	Random
Straight line	4.82	10.39	86.83
"A" with translation	3.62	14.81	104.56
"A" with zoom	1.71	19.58	84.67

Table 2. IMAGEBOT accuracy.

	Mean Error (mm)		
	Oracle	ARE	Random
Robot with camera	14.25	149.10	1482.83
Robot with laser	16.25	287.93	1450.50

Table 3. Mobile robot accuracy.

5.4 Robustness

Finally, we consider the robustness of this technique. The results in the previous section demonstrate one aspect of robustness—the ability to subjectively localize in two very different domains. Even more compelling, the primary sensor was switched from camera to laser and the robot was still able to successfully localize. The algorithm, entirely unchanged, found a new subjective representation, and new motion and sensor models without requiring time-consuming manual creation of these new models.

Parameters.

Another aspect is the robustness of the algorithm to the setting of its various parameters. There was no tuning of the parameters for any of the results presented here. All results used simple Euclidean distance as ARE's local distance metric over observations. All used a degree three polynomial when computing the sensor model. The final two parameters are the choice of the number of dimensions d in the subjective representation and the number of particles used in MCL. Varying the choice of d from two to eight dimensions in the IMAGEBOT line example affects the resulting accuracy by no more than 2 pixels. Varying the number of particles used in MCL from 50 to 500 caused no difference in the resulting trajectories. In summary, the presented technique has surprisingly few parameters and is quite robust to their choice.

Leaving the Map.

As a final consideration of robustness, an IMAGEBOT trajectory was examined where the test data included objective locations far outside the gathered training data. This means that the synthetic robot left its map for portions of its trajectory. The accuracy measure on the trajectory, averaged over ten runs, was 57 pixels, where the oracle was 37, and random was 158. The high errors for all techniques is due to the fact that for many test points no point in the training data was objectively close. Despite this, the subjective localization based on ARE continues to perform only marginally worse than the oracle.

6 Conclusion

In summary we examined the problem of subjective localization, where the algorithm can choose an appropriate frame of reference in which to localize. We proposed a technique for solving this problem by (i) extracting a subjective representation from training data using Action Respecting Embedding, (ii) learning a motion model and sensor model for this representation, and (iii) using these models with Monte Carlo localization to track the robot's location in the subjective frame of reference. We evaluated this technique in both a synthetic image manipulation domain and with a mobile robot. The algorithm successfully extracted subjective representations and localized on new test data with substantial accuracy. These results were consistent, with no changes to the algorithm, across a variety of different experiments, including changing the robot's primary sensor from camera to laser. We also showed that the algorithm was robust to the few parameters that it depends upon.

References

[BGW05] Michael Bowling, Ali Ghodsi, and Dana Wilkinson. Action respecting embedding. In *Proceedings of the Twenty-Second International Conference on Machine Learning*, pages 65–72, 2005.

[EP04] Austin I. Eliazar and Ronald Parr. Learning probabilistic motion models for mobile robots. In *Proceedings of the Twenty-First International Conference on Machine Learning*. ACM Press, 2004.

[FBDT99] D. Fox, W. Burgard, F. Dellaert, and S. Thrun. Monte carlo localization: Efficient position estimation for mobile robots. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence*, 1999.

[Kal60] R. E. Kalman. A new approach to linear filtering and prediction problems. *Journal of Basis Engineering*, pages 35–45, 1960.

[MTKW02] Michael Montemerlo, Sebastian Thrun, Koller Koller, and Ben Wegbreit. FastSLAM: A factored solution to the simultaneous localization and mapping problem. In *Proceedings of the Eighteenth National Conference on Artificial Intelligence*, pages 593–598, 2002.

[RT99] Nicholas Roy and Sebastian Thrun. Online self-calibration for mobile robots. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 1999.

[SC70] P. H. Schoenemann and R. Carroll. Fitting one matrix to another choice of a central dilation and a rigid motion. *Psychometrika*, 35:245–255, 1970.

[SS02] B. Scholkopf and A. Smola. *Learning with Kernels*. MIT Press, 2002.

[SS05] Daniel Stronger and Peter Stone. Simultaneous calibration of action and sensor models on a mobile robot. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 2005.

[WS04] K. Weinberger and L. Saul. Unsupervised learning of image manifolds by semidefinite programing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 988–995, 2004.