A Robust Autoregressive Gaussian Process Motion Model Using l_1-Norm Based Low-Rank Kernel Matrix Approximation

Eunwoo Kim, Sungjoon Choi, and Songhwai Oh

Abstract— **This paper considers the problem of modeling complex motions of pedestrians in a crowded environment. A number of methods have been proposed to predict the motion of a pedestrian or an object. However, it is still difficult to make a good prediction due to challenges, such as the complexity of pedestrian motions and outliers in a training set. This paper addresses these issues by proposing a robust autoregressive motion model based on Gaussian process regression using l_1-norm based low-rank kernel matrix approximation, called PCGP-l_1. The proposed method approximates a kernel matrix assuming that the kernel matrix can be well represented using a small number of dominating principal components, eliminating erroneous data. The proposed motion model is robust against outliers present in a training set and can reliably predict the motion of a pedestrian, such that it can be used by a robot for safe navigation in a crowded environment. The proposed method is applied to a number of regression and motion prediction problems to demonstrate its robustness and efficiency. The experimental results show that the proposed method considerably improves the motion prediction rate compared to other Gaussian process regression methods.**

I. Introduction

For service mobile robots assisting humans, adapting to complex and real-world environments is a challenging task [1]. The environments can be highly dynamic and it is difficult to predict behaviors of humans. In such environments, a service robot can easily get stuck or collide with moving humans or obstacles and it can lead to an unsafe situation for humans nearby. Since a safe operation is an important requirement for the success of service robots, an ability to predict motions of pedestrians and moving objects is of paramount importance.

In this paper, we consider the problem of safe navigation of a mobile robot under a dynamic and crowded environment [2]. A related topic is autonomous robot navigation and it has been studied extensively in recent years [3]–[10]. For safe navigation, it is required to predict the trajectories of pedestrians and moving objects in order to schedule a collision-free path for a robot. In [3], a probabilistic framework based on a partially observable Markov decision process is used to predict trajectories of humans and obstacles for autonomous robot navigation. Henry et al. [5] proposed a learning method for human-like navigation in a crowded environment using inverse reinforcement learning. Asaula et al. [6] proposed a stochastic abstraction of human motion using a discrete-time Markov chain to predict trajectories of moving objects and estimate the probability of an accident. Fulgenzi et al. [4] proposed a motion pattern model of pedestrians using a Gaussian process (GP). In many approaches, it is assumed that the current positions of a robot and dynamic obstacles are available [9] or positions can be estimated from an external device or infrastructure, such as an overhead camera network system [8]. However, such assumption makes an approach impractical in many real-world environments since collecting exact locations using an external device can be expensive and available only in a laboratory setting.

An autonomous navigation method can be grouped into four categories based on the perspective of a view (reference or egocentric) and the method used to predict future positions of obstacles (model-based or data-driven) [10]. A complex motion of a pedestrian is modeled using autoregressive Gaussian processes (AR-GP) in [10]. The AR-GP based method is data-driven and egocentric, hence, no external device is required for collecting location information. In [10], the authors have shown that the AR-GP based method gives a dramatic improvement over existing reactive control methods, such as the reactive planner [7] and vector field histogram [11]. However, in order for an AR-GP motion model to perform well, the training set has to be carefully collected such that no outliers are included. But it is a nontrivial task to collect an outlier-free training set due to noises in sensors and object detection algorithms used in a robot sensing system. In this paper, we address the limitation of [10] by introducing a method which can robustly learn from a training set with outliers.

To handle outliers, methods based on the l_1-norm are widely used to robustly solve problems such as background modeling, image denoising, dimension reduction, and data reconstruction [12], [13]. These techniques assume a Laplacian noise model instead of a Gaussian noise model and are used to obtain a robust low-rank approximation of the original data.

In this paper, we propose a robust autoregressive Gaussian process motion model (AR-GP) using the l_1-norm based low-rank kernel matrix approximation. The proposed method (PCGP-l_1) approximates a kernel matrix used in AR-GP by its low-rank approximation, assuming that the kernel matrix can be represented using a small number of dominating principal components, eliminating outliers and erroneous data in the training set. By extracting orthogonal basis vectors from l_1-norm based low-rank kernel matrix approximation, we

This research was supported in part by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (NRF-2013R1A1A2009348) and by the ICT R&D program of MSIP/IITP (14-824-09-014, Basic Software Research in Human-level Lifelong Machine Learning, Machine Learning Center).

E. Kim, S. Choi, and S. Oh are with the Department of Electrical and Computer Engineering, ASRI, Seoul National University, Seoul 151-744, Korea (e-mail: {eunwoo.kim, sungjoon.choi, songhwai.oh}@cpslab.snu.ac.kr).

obtain the robust solution of AR-GP. The proposed method is applied to regression and motion prediction problems in simulation to demonstrate its robustness against outliers. The method is also applied in a physical experiment using a Pioneer 3DX mobile robot and a Microsoft Kinect camera for motion prediction. We also applied the proposed motion model to a motion control problem in a school cafeteria. The experimental results show that our proposal is robust compared to existing Gaussian process regression methods under the existence of outliers or unwanted data.

The remainder of this paper is organized as follows: In Section II, we briefly review the Gaussian process motion model. In Section III, we proposed a robust autoregressive Gaussian process motion model using l_1-norm based low-rank kernel matrix approximation. In Section IV, the performance of our proposal is evaluated in experiments.

II. Autoregressive Gaussian Process Motion Model

A Gaussian process (GP) is a collection of random variables which has a joint Gaussian distribution and is specified by its mean function $m(\boldsymbol{x})$ and covariance function $k(\boldsymbol{x}, \boldsymbol{x}')$ [14]. A Gaussian process $f(\boldsymbol{x})$ is expressed as:

$$f(\boldsymbol{x}) \sim GP(m(\boldsymbol{x}), k(\boldsymbol{x}, \boldsymbol{x}')). \tag{1}$$

Suppose that $\boldsymbol{x}_i \in \mathbb{R}^{n_i}$ is an input and $y_i \in \mathbb{R}$ is an output. For a noisy observation set $D = \{(\boldsymbol{x}_i, y_i) | i = 1, \cdots, n\}$, we can consider the following observation model:

$$y_i = f(\boldsymbol{x}_i) + w_i, \tag{2}$$

where $w_i \in \mathbb{R}$ is a zero-mean white Gaussian noise with variance σ_w^2. Then the covariance of y_i and y_j can be expressed as

$$cov(y_i, y_j) = k(\boldsymbol{x}_i, \boldsymbol{x}_j) + \sigma_w^2 \delta_{ij}, \tag{3}$$

where δ_{ij} is the Kronecker delta function which is 1 if $i = j$ and 0 otherwise. $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \langle \phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j) \rangle$ is a covariance function based on some nonlinear mapping function ϕ. The function k is also known as a kernel function.

We can represent (3) in a matrix form as follows:

$$cov(\boldsymbol{y}) = K + \sigma_w^2 I, \tag{4}$$

where $\boldsymbol{y} = [y_1 \cdots y_n]^T$ and K is a kernel matrix such that $[K]_{ij} = k(x_i, x_j)$.

The conditional distribution of a new output y_* at a new input $\boldsymbol{x}_*$ given D becomes

$$y_* | D, \boldsymbol{x}_* \ \sim \ \mathcal{N}(\overline{y}_*, \mathbb{V}(y_*)), \tag{5}$$

where

$$\overline{y}_* = k_*^T (K + \sigma_w^2 I)^{-1} \boldsymbol{y} \tag{6}$$

and the covariance of y_*, $\mathbb{V}(y_*)$, is

$$\mathbb{V}(y_*) = k(\boldsymbol{x}_*, \boldsymbol{x}_*) - k_*^T (K + \sigma_w^2 I)^{-1} k_*. \tag{7}$$

Here, $k_* \in \mathbb{R}^n$ is a covariance vector between the new data $\boldsymbol{x}_*$ and the existing data, such that $[k_*]_i = k(x_*, x_i)$.

In [10], Choi et al. proposed an autoregressive Gaussian process (AR-GP) for motion modeling and prediction of a pedestrian trajectory for real-time robot navigation. Let $D_t = (x_t, y_t) \in \mathbb{R}^2$ be the position of a moving object at time t, they modeled the current velocity, $\Delta D_t = D_t - D_{t-1}$, with an appropriate time scaling using the AR-GP model as follows [10]:

$$\begin{aligned} \Delta D_t &= f(D_{t-1}, D_{t-2}, ..., D_{t-p}) \\ &\sim GP_f(D_{t-1}, D_{t-2}, ..., D_{t-p}). \end{aligned} \tag{8}$$

Hence, the AR-GP model approximates the current motion of a moving object from p recent positions of the object. It can be considered as a nonlinear generalization of an autoregressive model under the Gaussian process framework. They have shown that an AR-GP based method gives a dramatic improvement compared with other reactive control algorithms, such as the reactive planner [7] and vector field histogram [11].

While Gaussian process regression is a nonparametric method which enjoys the expressive power of the full Bayesian framework, it suffers from a major drawback which is its high computational cost for evaluating the inverse of a kernel matrix whose size grows with the number of training data [14]. To deal with the computation issue, sparse Gaussian process methods have been proposed [15], [16]. These methods give results comparable to the standard GP but with lower computational complexity. However, note that when it comes to make a prediction from a fixed training set, the computational load of calculating (6) and (7) can be reduced by pre-computing the inverse of the kernel matrix as follows [10]:

$$\overline{y}_* = k_*^T \Lambda \boldsymbol{y} = k_*^T \Gamma, \tag{9}$$

where $\Lambda = (K + \sigma_w^2 I)^{-1}$ and $\Gamma = \Lambda \boldsymbol{y}$.

III. Robust Autoregressive Gaussian Process Motion Model

A. Principal Component Gaussian Process (PCGP)

To reduce the computational cost of inverting the kernel matrix Λ in (9), a number of approximation methods have have been proposed, including Incomplete Cholesky Factorization (ICF) [16] and the Nyström method [17]. In this paper, we consider another kernel matrix approximation method, low-rank kernel matrix approximation, which is also known as kernel principal component analysis (KPCA) [18]. It has been attracted much attention for a wide range of problems in order to efficiently process a large quantity of data and to discover a hidden low-dimensional structure based on the Euclidean distance (l_2-norm).

The main idea behind the kernel-based approximation method is that by using a kernel function the original linear operations of PCA are performed in a high-dimensional Hilbert space [18]. Performing linear PCA in a high-dimensional space has an effect of performing nonlinear PCA in the original input space [18]. Hence, we can apply low-rank kernel matrix approximation to reduce the computational load of Λ in (9) to speed up the kernel machine.

Suppose that a nonlinear function $\phi : \mathbb{R}^{n_i} \rightarrow \mathbb{X}$ is a mapping from the input space to a high-dimensional feature space. Then, for centered data $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$, the covariance matrix in $\mathbb{X}$ is

$$C = \frac{1}{n} \sum_{i=1}^{n} \phi(\boldsymbol{x}_i) \phi(\boldsymbol{x}_i)^T$$

and the eigenvector $\boldsymbol{v}$ with nonzero eigenvalue of C can be represented as $\boldsymbol{v} = \sum_{i=1}^{n} \beta_i \phi(\boldsymbol{x}_i)$. The coefficients $\boldsymbol{\beta} = [\beta_1 \cdots \beta_n]^T$ can be found by solving the following eigenvalue problem [18]:

$$K\boldsymbol{\beta} = n\lambda\boldsymbol{\beta}, \tag{10}$$

where K is a kernel matrix such that $[K]_{ij} = \langle \phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j) \rangle$. It follows that principle components in $\mathbb{X}$ can be extracted using eigenvectors $\boldsymbol{v}_k$ for $k = 1, \ldots, r$ with r largest eigenvalues of K using coefficents found from (10) with a proper normalization [18]. Hence, we can effectively represent a kernel matrix using a subset of eigenvectors with r largest eigenvalues.

Given the eigenvalue decomposition of $K = R\Sigma R^T$, where $\Sigma = \text{diag}(\lambda_1, \cdots, \lambda_n)$ is a diagonal matrix of eigenvalues of K, such that $\lambda_1 \geq \cdots \geq \lambda_n$, we can approximate the inverse of K as follows:

$$K^{-1} = (R\Sigma R^T)^{-1} = R\Sigma^{-1}R^T \approx \widetilde{R}\widetilde{R}^T, \tag{11}$$

where $\widetilde{R} = R_r \Sigma_r^{-\frac{1}{2}}$. Here, $R_r \in \mathbb{R}^{n \times r}$ collects the first r vectors from R and $\Sigma_r = \text{diag}(\lambda_1, \cdots, \lambda_r) \in \mathbb{R}^{r \times r}$ is a diagonal matrix of r largest eigenvalues. Hence, we can reformulate (9) by treating Λ as K in (11) as

$$\bar{\boldsymbol{y}}_* = k_*^T \Lambda \boldsymbol{y} \approx k_*^T \widetilde{R}\widetilde{R}^T \boldsymbol{y} = \widetilde{k}_*^T \widetilde{\boldsymbol{y}}, \tag{12}$$

where $\widetilde{k}_* = \widetilde{R}^T k_*$ is a kernel vector which is projected into the orthogonal feature space $\widetilde{R}$ and $\widetilde{\boldsymbol{y}} = \widetilde{R}^T \boldsymbol{y}$ is a projected output vector into $\widetilde{R}$. This means that the low-dimensional approximation of a kernel matrix can be applied to Gaussian process regression problems by using $\widetilde{k}_*$ and $\widetilde{\boldsymbol{y}}$ which are projected on $\widetilde{R}$, and the inverse of kernel matrix becomes an identity matrix which represents the independent relationship between basis vectors. Hence, (12) can be another representation of $\overline{y}_*$ in the dimensionally reduced orthogonal feature space $\widetilde{R}$. Figure 1 shows the concept of the proposed method using low-rank kernel matrix approximation.

B. PCGP-l_1: PCGP via l_1-norm based kernel matrix approximation for robust AR-GP

As mentioned earlier, Λ in (9) can be approximated by a conventional low-rank approximation method which transforms data into a low-dimensional subspace and maximizes the variance of the given data based on the Euclidean distance (l_2-norm). However, the method is sensitive to outliers because the l_2-norm can sometimes amplify the negative effects of such data. When outliers are present in the data, the covariance matrix of the data can be corrupted by outliers. Therefore, l_2-norm based low-rank approximation methods may find projections which are far from the desired solution. As an alternative, various methods based on the l_1-norm have been proposed recently, which are more robust against outliers [13]. Hence, we approximate the kernel matrix Λ using l_1-norm based kernel matrix approximation for robust autoregressive Gaussian process motion model learning.

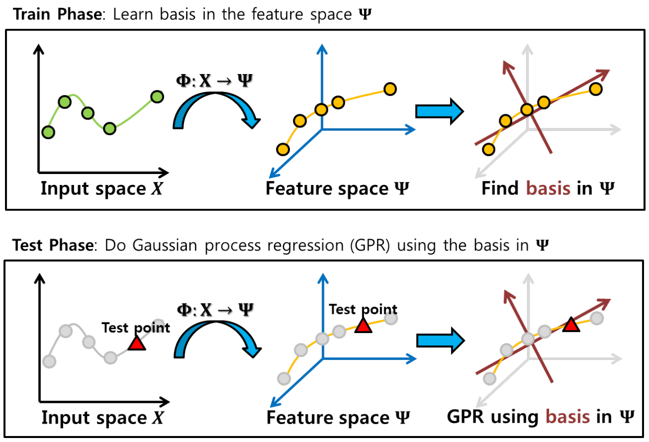


Fig. 1. A graphical illustration of principal component Gaussian process regression using low-rank kernel matrix approximation. We can perform the prediction step of Gaussian process regression in the dimension-reduced feature space.

A minimization problem based on the l_1-norm can be regarded as a maximum likelihood estimation problem under the Laplacian noise distribution [13]. We first consider an approximation problem for vector $\boldsymbol{g} = (g_1, g_2, \cdots, g_m)^T$ by a multiplication of vector $\boldsymbol{x} \in \mathbb{R}^m$ and scalar α, i.e.,

$$\boldsymbol{g} = \alpha\boldsymbol{x} + \boldsymbol{\delta}, \tag{13}$$

where $\boldsymbol{\delta}$ is a noise vector whose elements are independently and identically distributed from a Laplacian distribution. The probability model for (13) can be written as

$$p(\boldsymbol{g}|\boldsymbol{x}) \sim \exp\left(-\frac{\|\boldsymbol{g} - \alpha\boldsymbol{x}\|_1}{s}\right), \tag{14}$$

where $\|\cdot\|_1$ denotes the l_1-norm and $s > 0$ is a scaling constant. We assume for a moment that $\boldsymbol{x}$ is given. In this case, maximizing the log likelihood of the observed data is equivalent to minimizing the following cost function:

$$J(\alpha) = \|\boldsymbol{g} - \alpha\boldsymbol{x}\|_1. \tag{15}$$

Let us rewrite the cost function for matrix approximation using (15):

$$\min_{U,V} \quad J(U, V) = \|G - UV\|_1. \tag{16}$$

where $G \in \mathbb{R}^{m \times n}$, $U \in \mathbb{R}^{m \times r}$, and $V \in \mathbb{R}^{r \times n}$ are the observation, projection, and coefficient matrices, respectively. G is a given data matrix and it is Λ from (12) for the low-rank kernel matrix approximation problem, i.e., $m = n$. Here, r is a predefined parameter less than $\min(m, n)$ and $\|\cdot\|_1$ is the entry-wise l_1-norm, i.e., $\|G\|_1 = \sum_{ij} |g_{ij}|$ where g_{ij} is the (i, j)-th element of G. In general, (16) is a non-convex problem, because both U and V are unknown variables.

We can efficiently perform this optimization process by minimizing the cost function over one matrix while keeping the other fixed and then alternately exchanging roles of matrices. Such minimization technique based on alternating iterations has been widely used in subspace analysis [13] and can be written as

$$\begin{aligned} U^{(j)} &= \arg\min_{U} \|G - UV^{(j-1)}\|_1 \\ V^{(j)} &= \arg\min_{V} \|G - U^{(j)}V\|_1, \end{aligned} \tag{17}$$

where the superscript j denotes the iteration number. However, conventional alternating minimization based approaches are still computationally very expensive and require a large memory when the matrix dimension is large.

In order to speed up the optimization step (17), [13] proposed two l_1-norm based low-rank approximation methods using alternating rectified gradient. In this paper, we use the second method (l_1-ARG$_D$) which solves the problem by optimizing its dual problem which is convex [13]. In addition, l_1-ARG$_D$ enjoys the convergence guarantee [13]. The problem of finding the gradient of V for a fixed U in l_1-ARG$_D$ is formulated as

$$\begin{aligned} \min_{\Delta V} \quad & J(\Delta V|U,V) = \|G - U(V+\Delta V)\|_1 \\ s.t. \quad & ||U\Delta V||_F^2 = \epsilon^2, \end{aligned} \tag{18}$$

where ΔV is the variation of V that we are seeking. In this formulation, l_1-ARG$_D$ solves the problem based on the primal-dual formulation and uses the proximal gradient and QR factorization for exact and fast computation. The computational complexity of l_1-ARG$_D$ at each iteration is $O(rmn)$ for a matrix of size $m\times n$ and is $O(rn^2)$ for a kernel matrix of size $n\times n$. l_1-ARG$_D$ typically requires between 10 and 20 iterations before convergence [13]. It has shown the superiority of l_1-ARG$_D$ for large-scale problems with respect to the running time and reconstruction errors compared with other l_1-norm based methods [13].

The overall procedure of the proposed robust Gaussian process regression is described in Algorithm 1. We call the algorithm as PCGP-l_1 because the algorithm uses principal components obtained from l_1-norm based low-rank kernel matrix approximation. In Algorithm 1, we perform the standard PCA to L, which is a low-rank kernel matrix, because it is difficult to find an exact solution with the same orthogonal U and V using the l_1-norm based method. We precompute the kernel matrix and its principal components in line 4-8, and test a new input $\boldsymbol{x}_*$ given the principal components R in line 10-11.

IV. Experimental Results

In this section, we have evaluated the performance of the proposed method (PCGP-l_1) by experimenting with various data and compared with well-known sparse Gaussian process methods, such as SPGP[1] [15] and GPLasso[2] [16], along with the standard full GP. In our experiments, we used the RBF Gaussian kernel for all methods and hyperparameters learned using a conjugate gradient method [14].

Algorithm 1 PCGP-l_1

```
 1: Input: $X, \boldsymbol{y}$, and $\boldsymbol{x}_*$
 2: Output: $\overline{\boldsymbol{y}}_*$
 3: // Training
 4:     Compute $\Lambda = K + \sigma_w^2 I$
 5:     Perform low-rank kernel matrix approximation [13]:
 6:         $[U,V] = \arg\min_{U,V} ||\Lambda - UV||_1$
 7:     $L \leftarrow UV$
 8:     Compute $R$ and $\Sigma$ by performing PCA to $L$
 9: // Testing
10:     Compute $k_* = k(\boldsymbol{x}_*, X)$
11:     Compute $\overline{\boldsymbol{y}}_*$ by (12)
```

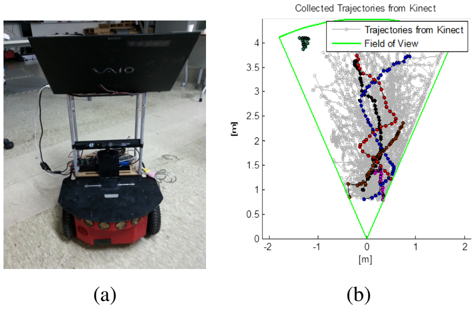


Fig. 2. (a) A Pioneer 3DX mobile robot with a Kinect camera. (b) Collected trajectories from Kinect. We show a few trajectories in bold for better visualization.

For an actual experiment, we collected trajectories of moving pedestrians using a Pioneer 3DX differential drive mobile robot and a Microsoft Kinect camera, which is mounted on top of the robot as shown in Figure 2(a). All algorithms are written in MATLAB using the mex-compiled ARIA package[3] on a notebook with a 2.1 GHz quad-core CPU and 8GB RAM. The position of a pedestrian is detected using the skeleton grab API for Kinect.

A. Simulation

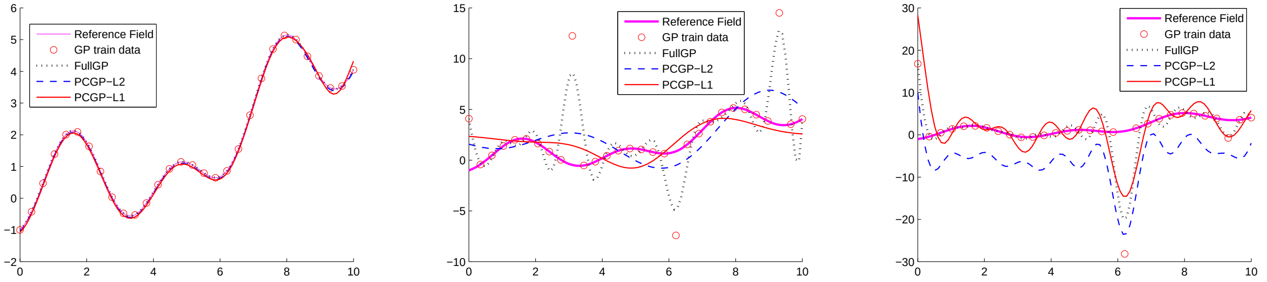
First, we tested on a synthetic example to compare the proposed l_1-norm based PCGP to the l_2-norm based PCGP in a regression problem when there are some outliers. Since we are interested in how the proposed PCGP-l_1 performs in the presence of outliers, we only compare PCGP-l_1 to l_2-norm based PCGP and the full GP.

Figure 3 shows the simulation results of a regression problem according to outlier levels. Two cases with outliers are constructed by adding additional outliers (10%) to the no outlier case with two different magnitude levels. As shown in Figure 3(a), all methods give the exact results when there are no outliers, so the proposed method shows its competitiveness compared with the full GP. If we add some outliers as shown in Figure 3(b), the full GP and PCGP-l_2 try to fit outliers so they show fluctuations when there is an outlier, but PCGP-l_1 is less affected by outliers, showing

[1] Available at http://www.gatsby.ucl.ac.uk/~snelson/.

[2] Available at https://www.cs.purdue.edu/homes/alanqi/softwares/softwares.htm.

[3] Available at http://robots.mobilerobots.com/wiki/ARIA.



(a)

(b)

(c)

Fig. 3. Simulation results on a synthetic example. (a) No outlier case. (b) Mid-level outlier case. (c) High-level outlier case.

its robustness against outliers. When there are outliers with larger magnitudes (Figure 3(c)), l_2-norm based PCGP tries to fit the outliers which results in a bad regression result. Unlike PCGP-l_2, PCGP-l_1 gives the best regression result. From this experiment, we see the clear benefit of the proposed method to a regression problem when the train set contains outliers.

B. Motion Prediction

We performed experiments in our laboratory to predict the future position of a person. From training, we collected a diverse set of trajectories of pedestrians, which are in the field of view of a robot as shown in Figure 2(b). To make a training set from the collected trajectories, we uniformly sampled positions to have about ten samples in a trajectory when a trajectory has many detected positions.

We evaluated the proposed method under this experimental setting compared with other Gaussian process methods (full GP [10], SPGP [15], and GPLasso [16]). To test the proposed method, we predicted the next location of a pedestrian given the detected locations of the moving pedestrian and this process is continuously performed until the pedestrian disappears from the field of view of a robot. We tested about 100 trajectories of pedestrians from a mobile robot in our laboratory and measured the prediction accuracy using the root mean squared error.

Figure 4 shows the motion prediction results in centimeters under various low rank conditions, $r/n = \{5\%, 10\%, 15\%\}$, using Kinect-based pedestrian trajectories. From the figure, we can see that the proposed method gives a lower error than other methods regardless of the rank condition, whereas the sparsity-based GP methods shows a higher root mean square error (RMSE) compared to the propose method and full GP. Some snapshots from our laboratory experiments are shown in Figure 5. For this example, the reduced rank was $r = 0.15n$. As shown in the figure, the proposed method predicted the future trajectories better than the full GP [10].

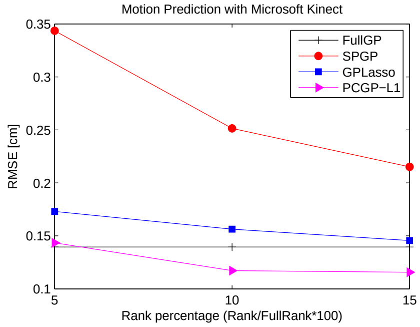


Fig. 4. Motion prediction results using Kinect-based human trajectories.

C. Motion Control

We evaluated the performance of the proposed method, PCGP-l_1, for a motion control problem. A non-parametric Bayesian motion controller can navigate through crowded dynamic environments using an autoregressive Gaussian process model (AR-GP). Thus, we applied the proposed method to the autoregressive Gaussian process motion controller [10]. The number of pedestrians varies from one to eight to validate the safety of the the proposed method under different conditions. Similar to [10], the robot tries to reach the goal using the pure pursuit method [19] if there is no pedestrian.

We tested the motion controller based on the proposed motion model in a crowded school cafeteria. The goal of the experiment was to follow a given path while avoiding dynamic or static pedestrians. Figure 6 shows some snapshots from the experiment. The robot successfully avoided moving pedestrians without any collisions and arrived at the goal position.

V. Conclusions

In this paper, we have proposed a robust Gaussian process regression method using l_1-norm based low-rank kernel matrix approximation, PCGP-l_1, for robustly modeling a complex pedestrian motion pattern in the presence of outliers. Using the relationship between Gaussian process regression and low-rank kernel matrix approximation, we obtain orthogonal basis vectors under the l_1-norm cost function. The proposed method is applied to various problems including experiments using a Pioneer 3DX mobile robot and a Microsoft Kinect. The experimental results show the performance and robustness of the proposed method against outliers and sensor errors compared to existing methods.

References

[1] C. H. Chen, Y. Weng, and S.-T. Sun, “Toward the human-robot coexistence society: on safety intelligence for next generation robots,”

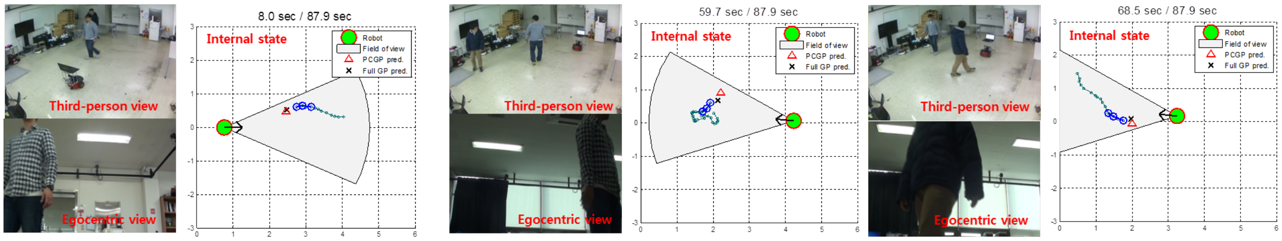


Fig. 5. Snapshots from laboratory experiments. The predicted positions by the proposed algorithm and the full GP motion model [10] are also shown. Each snapshot shows a photo taken by a camera, a photo taken by the robot, and the robot's internal state.

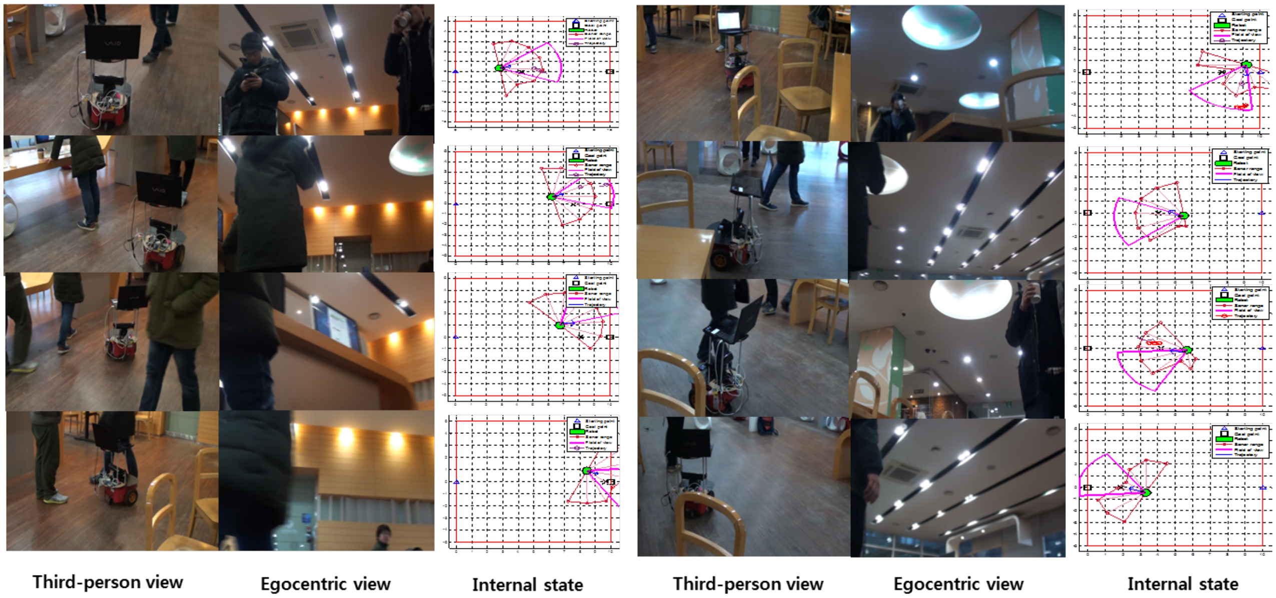


Fig. 6. Snapshots from the school cafeteria experiment using the autoregressive Gaussian process motion controller based on PCGP-l_1. Each snapshot shows a photo taken by a camera, a photo taken by the robot, and the robot's internal state.

International Journal of Social Robotics, vol. 1, no. 4, pp. 267–282, 2009.

[2] D. Helbing and P. Molnár, "Social force model for pedestrian dynamics," *Physical Review E*, vol. 51, no. 5, pp. 4282–4286, 1995.

[3] A. F. Foka and P. E. Trahanias, "Predictive autonomous robot navigation," in *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2002.

[4] C. Fulgenzi, C. Tay, A. Spalanzani, and C. Laugier, "Probabilistic navigation in dynamic environment using rapidly-exploring random trees and Gaussian processes," in *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2008.

[5] P. Henry, C. Vollmer, B. Ferris, and D. Fox, "Learning to navigate through crowded environments," in *Proc. of IEEE International Conference on Robotics and Automation (ICRA)*, 2010.

[6] R. Asaula, D. Fontanelli, and L. Palopoli, "Safety provisions for human/robot interactions using stochastic discrete abstractions," in *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2010.

[7] J. J. Park, C. Johnson, and B. Kuipers, "Robot navigation with model predictive equilibrium point control," in *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012.

[8] P. Trautman, J. Ma, R. M. Murray, and A. Krause, "Robot navigation in dense human crowds: the case for cooperation," in *Proc. of IEEE International Conference on Robotics and Automation (ICRA)*, 2013.

[9] G. S. Aoude, B. D. Luders, J. M. Joseph, N. Roy, and J. P. How, "Probabilistically safe motion planning to avoid dynamic obstacles with uncertain motion patterns," *Autonomous Robots*, no. 1, pp. 51–76, 2013.

[10] S. Choi, E. Kim, and S. Oh, "Real-time navigation in crowded dynamic environments using Gaussian process motion control," in *Proc. of IEEE International Conference on Robotics and Automation (ICRA)*, 2014.

[11] I. Ulrich and J. Borenstein, "VFH*: local obstacle avoidance with look-ahead verification," in *Proc. of IEEE International Conference on Robotics and Automation (ICRA)*, 2000.

[12] E. J. Candes, X. Li, Y. ma, and J. Wright, "Robust principal component analysis?" *Journal of the ACM*, vol. 58, pp. 11:1–11:37, 2011.

[13] E. Kim, M. Lee, C.-H. Choi, N. Kwak, and S. Oh, "Efficient l_1-norm-based low-rank matrix approximations for large-scale problems using alternating rectified gradient method," *IEEE Trans. on Neural Networks and Leraning Systems (TNNLS)*, 2014 (Online).

[14] C. Rasmussen and C. Williams, *Gaussian process for machine learning*. MIT Press, 2006.

[15] E. Snelson and Z. Ghahramani, "Sparse Gaussian processes using pseudo-inputs," in *Advances in Neural Information Processing Systems (NIPS)*, 2005.

[16] F. Yan and Y. Qi, "Sparse Gaussian process regression via l_1 penalization," in *Proc. of the International Conference on Machine Learning (ICML)*, 2010.

[17] C. Williams and M. Seeger, "Using the Nyström method to speed up kernel machines," in *Advances in Neural Information Processing Systems (NIPS)*, 2001.

[18] B. Scholkopf, A. Smola, and K.-R. Muller, "Nonlinear component analysis as a kernel eigenvalue problem," *Neural Computation*, vol. 10, pp. 1299–1319, 1998.

[19] O. Amidi, "Integrated mobile robot control," Robotics Institute, Pittsburgh, PA, Tech. Rep. CMU-RI-TR-90-17, May 1990.