Leveraged Non-Stationary Gaussian Process Regression for Autonomous Robot Navigation

Sungjoon Choi, Eunwoo Kim, Kyungjae Lee, and Songhwai Oh

Abstract— **In this paper, we propose a novel regression method that can incorporate both positive and negative training data into a single regression framework. In detail, a leveraged kernel function for non-stationary Gaussian process regression is proposed. With this new kernel function, we can vary the correlation betwen two inputs in both positive and negative directions by adjusting leverage parameters. By using this property, the resulting leveraged non-stationary Gaussian process regression can anchor the regressor to the positive data while avoiding the negative data. We first prove the positive semi-definiteness of the leveraged kernel function using Bochner's theorem. Then, we apply the leveraged non-stationary Gaussian process regression to a real-time motion control problem. In this case, the positive data refer to *what to do* and the negative data indicate *what not to do*. The results show that the controller using both positive and negative data outperforms the controller using positive data only in terms of the collision rate given training sets of the same size.**

I. Introduction

Recently, machine learning techniques using kernel methods, such as kernel support vector machines (SVMs), kernel principal component analysis (KPCA), and Gaussian processes have gained popularity. Mathematically, a kernel function $k(\mathbf{x},\mathbf{x}') : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is equivalent to an inner-product, $\langle\langle \phi(\mathbf{x}), \phi(\mathbf{x}') \rangle\rangle_{\mathcal{H}}$, in the reproducing kernel Hilbert space (RKHS), $\mathcal{H}$, with some mapping function $\phi : \mathcal{X} \to \mathcal{H}$. The existence of the mapping ϕ can be shown using the Mercer's theorem which states that if a kernel function $k(\cdot,\cdot)$ is positive semi-definite, there exists a corresponding RKHS emphasizing the importance of the positive semi-definiteness of a kernel function. Intuitively speaking, a kernel function $k(\mathbf{x},\mathbf{x}')$ indicates a correlation between two input points $\mathbf{x}$ and $\mathbf{x}'$. Understandably, defining an appropriate kernel function is of the utmost importance in using kernel tricks and numerous kernels have been proposed for diverse purposes [1], [2].

In particular, Gaussian process regression (GPR) has been widely used in a number of practical applications [3], [4]. Gaussian process regression is a non-parametric Bayesian regression algorithm which predicts the output of an unseen input by assuming both training data and unseen input follow a jointly Gaussian distribution. In this case, a kernel function $k(\mathbf{x},\mathbf{x}')$ defines the covariance between $\mathbf{x}$ and $\mathbf{x}'$. Since a covariance matrix K, whose entries are specified by the kernel function, i.e., $K_{ij} = k(x_i, x_j)$, must satisfy positive semi-definiteness, a kernel function for a Gaussian process must be a positive semi-definite function.

Since a kernel function indicates the correlation between two input data, most of the existing kernel functions are decreasing functions of the distance between inputs. Perhaps, the most famous one is a squared exponential (SE) kernel function $k(\mathbf{x},\mathbf{x}') = \sigma \exp\left(-\frac{\|\mathbf{x}-\mathbf{x}'\|^2}{2l}\right)$. To the best of our knowledge, the only exceptions are the periodic kernels [1], [5]. In a random process, when the covariance between two inputs, $\mathbf{x}$ and $\mathbf{x}'$, are completely specified by a function of displacement between two inputs, $\mathbf{x} - \mathbf{x}'$, we call such processes and covariance functions as stationary random processes and stationary covariance functions, respectively. When these stationary covariance functions are used in Gaussian process regression, we call it stationary GPR.

The assumption behind stationary GPR is that the function we are trying to model has a *homogeneous* smoothness. In order to handle functions with *inhomogeneous* smoothness, non-stationary GPR has been proposed [6]. The main idea is to add additional parameters in a kernel function for controlling the local smoothness while preserving the positive semi-definiteness. In this case, however, one must define a smoothness over all input space, which often requires heavy computational load [6].

In this paper, we propose a leveraged non-stationary kernel function which can control the *leverage* of each training data, not only to the *positive* direction, but also to the *negative* direction while satisfying the positive semi-definiteness condition. In the general regression framework, the training data, $D = \{(\mathbf{x}, y)_i \,|\, i = 1, ..., N\}$, where $\mathbf{x}_i \in \mathcal{X}$ is an input and $y_i \in \mathcal{Y}$ is an output, usually work as anchor points, making the regressor $\hat{f} : \mathcal{X} \to \mathcal{Y}$ to be *close* to points in D. The proposed kernel function, however, let the regression result to be close to *positive* training points and far from *negative* training points. When this regression framework is adapted to a motion controller, the *positive* data and *negative* data indicate *'what to do'* and *'what not to do'*, respectively. From various robot navigation simulations, controlling with both positive and negative examples is much more effective and efficient than controlling with positive examples only in terms of the required number of data.

The remaining paper is organized as follows. In Section II, non-stationary Gaussian process regression is introduced. A new regression framework that can use both positive and negative training data is proposed in Section III. In Section IV and V, we apply the proposed non-stationary Gaussian

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).

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

process regression to the motion control problem and validate the performance both in simulations and actual experiments using a Pioneer 3DX mobile robot with two Kinect cameras.

II. Preliminaries

In probability theory, a random process is called a wide-sense stationary process if its mean function $m(\mathbf{x})$ is constant and a covariance function $k(\mathbf{x},\mathbf{x}')$ is a function of a displacement vector $k(\mathbf{x},\mathbf{x}') = k_S(\mathbf{x}-\mathbf{x}')$. If a covariance function is a function of the distance between two inputs, $k(\mathbf{x},\mathbf{x}') = k_I(\|\mathbf{x}-\mathbf{x}'\|)$, such covariance function is called an *isotropic* kernel function.

Gaussian process regression (GPR) is called stationary GPR if a stationary kernel function is used. The assumption behind using stationary GPR is that the function of our interest has the homogeneous smoothness. However, this may not be always the case, and in [6], the authors proposed non-stationary Gaussian process regression using the following non-stationary kernel function:

$$k(\mathbf{x}_i,\mathbf{x}_j) = \frac{2^{\frac{p}{2}}|\Sigma_i|^{\frac{1}{4}}\Sigma_j|^{\frac{1}{4}}}{|\Sigma_i+\Sigma_j|^{\frac{1}{2}}}\exp\left(-(\mathbf{x}_i-\mathbf{x}_j)^T\bar{\Sigma}^{-1}(\mathbf{x}_i-\mathbf{x}_j)\right). \tag{1}$$

With this non-stationary kernel function, we can manually control the local smoothness by additional parameters: Σ_i and Σ_j. Since a kernel matrix works as a covariance matrix in a Gaussian process, it must satisfy the positive semi-definite condition:

$$\sum_{i=1}^{N}\sum_{j=1}^{N} a_i a_j C(\mathbf{x}_i,\mathbf{x}_j) \geq 0, \tag{2}$$

for every $\mathbf{a} = [a_1 \ \dots \ a_N]^T \in \mathbb{R}^N$ and N.

Unfortunately, directly showing (2) for a given kernel function is not trivial. In [6], the authors proved the positive semi-definiteness of (1) using the following theorem by Higdon, Swall, and Kern (HSK) [7].

Theorem 1 (HSK Theorem): A covariance function $C(x_i,x_j)$ defined by

$$C(\mathbf{x_i},\mathbf{x_j}) = \int_{\mathbb{R}^d} K_{\mathbf{x_i}}(\mathbf{u})K_{\mathbf{x_j}}(\mathbf{u})d\mathbf{u}, \tag{3}$$

where $\mathbf{x_i}$, $\mathbf{x_j}$, $\mathbf{u} \in \mathbb{R}^d$, and $K_{\mathbf{x}}(\cdot)$ is a basis kernel function centered at $\mathbf{x}$, is positive semi-definite.

Using Theorem 1, one can avoid the difficulties in verifying the positive semi-definiteness of a kernel function. Deriving the non-stationary kernel function (1) can be easily done by setting $K_{\mathbf{x}}(\mathbf{u}) = \mathcal{N}(\mathbf{u}|\mathbf{x},\Sigma)$. The integration over $\mathbb{R}^d$ can be handled using the properties of Gaussian distributions, in that the product of two Gaussian probability density functions is a pseudo-Gaussian.

However, it is still difficult to prove positive semi-definiteness of an arbitrarily given function as we have to find a basis kernel $K_{\mathbf{x}}(\mathbf{u})$ which forms the given function of interest as shown in (3). Another way of showing positive semi-definiteness is using the Bochner's theorem [8].

Theorem 2 (Bochner's Theorem): Let f be a bounded continuous function on $\mathbb{R}^d$. Then, f is positive semi-definite if and only if it is the (inverse) Fourier transform of a nonnegative and finite Borel measure μ, i.e.,

$$f(\mathbf{x}) = \int_{\mathbb{R}^d} e^{i\mathbf{w}^T\mathbf{x}}\mu(d\mathbf{w}). \tag{4}$$

In particular, Theorem 2 states that if a Fourier transform of a function f is non-negative, then f is positive semi-definite.

Theorem 3: If a bounded continuous function $f:\mathbb{R}^d \to \mathbb{R}$ is a (inverse) Fourier transform of a non-negative density function of a Borel measure μ, then f is positive semi-definite.

Proof: Let $g(\mathbf{w}) \geq 0$ be a density of a Borel measure μ, i.e.,

$$\mu(E) = \int_E g(\mathbf{w})d\mathbf{w}.$$

If we assume g is a Fourier transform of a bounded and continuous function f, f can be written as follows:

$$f(\mathbf{x}) = \int_{\mathbb{R}^d} e^{j\mathbf{w}^T\mathbf{x}}g(\mathbf{w})d\mathbf{w}. \tag{5}$$

From (5), one can directly prove positive semi-definiteness of f as follows:

$$\begin{aligned}
&\sum_{n=1}^{N}\sum_{m=1}^{N} a_n a_m^* f(\mathbf{x}_n-\mathbf{x}_m) \\
&= \sum_{n=1}^{N}\sum_{m=1}^{N} a_n a_m^* \int_{\mathbb{R}^d} e^{j\mathbf{w}^T(\mathbf{x}_n-\mathbf{x}_m)}g(\mathbf{w})d\mathbf{w} \\
&= \int_{\mathbb{R}^d}\sum_{n=1}^{N}\sum_{m=1}^{N} a_n a_m^* e^{j\mathbf{w}^T(\mathbf{x}_n-\mathbf{x}_m)}g(\mathbf{w})d\mathbf{w} \\
&= \int_{\mathbb{R}^d}\left(\sum_{n=1}^{N} a_n e^{j\mathbf{w}^T\mathbf{x}_n}\right)\left(\sum_{m=1}^{N} a_m^* e^{-j\mathbf{w}^T\mathbf{x}_m}\right)g(\mathbf{w})d\mathbf{w} \\
&= \int_{\mathbb{R}^d}\left\|\sum_{n=1}^{N} a_n e^{j\mathbf{w}^T\mathbf{x}_n}\right\|^2 g(\mathbf{w})d\mathbf{w} \quad \geq 0.
\end{aligned}$$

■

This theorem will be used in Section III-A to prove positive semi-definiteness of the proposed kernel function.

III. Regression with Both Positive and Negative Examples

The non-stationarity of the kernel function (1) comes from the kernel's capability to adjust local smoothness by assigning Σ_i for each data sample $\mathbf{x}_i$. In this section, we focus on the leveraged non-stationary kernel function for a non-stationary Gaussian process that can vary the *leverage* of training data.

The motivation behind the development of this kernel function starts with a relatively simple question. *Is it possible to use negative examples in a regression framework?* As the goal of a traditional regression problem is to find a function (or a regressor) that can best *fit* given training data, $D = \{(\mathbf{x},y)_i, i = 1,...,N\}$, where $\mathbf{x}_i$ and y_i are input and output, respectively, it usually anchor the regressor to those input and output points.

In our regression formulation, *positive* data work as an attractive force making the regressor as close as possible to such points, and *negative* data work as a repulsive force making the regressor as far as possible from such points. Moreover, it can also vary the *leverage* of each training data from fully negative (-1) to fully positive ($+1$). In particular, when the *leverage* is 0, the corresponding data will have no effect on the regression.

A. Leveraged Kernel Function

In this section, we propose a leveraged kernel function that can be used in a novel regression framework which can incorporate both positive and negative training data. Furthermore, we prove that the proposed leveraged non-stationary kernel function satisfies the positive semi-definiteness (PSD) condition using kernel composite rules and Theorem 3.

Each training data has its *leverage* parameter which varies from -1 to $+1$, where -1 indicates a fully negative leverage and $+1$ indicates a fully positive leverage. Following is the proposed leveraged kernel function:

$$k(\mathbf{x}_i, \mathbf{x}_j) = (1 - |\gamma_i - \gamma_j|)\, k_{SE}(\mathbf{x}_i, \mathbf{x}_j), \tag{6}$$

where $k_{SE}(\mathbf{x}_i, \mathbf{x}_j) = \sigma_f^2 \exp\left(-\frac{1}{2}(\frac{\|\mathbf{x}_i - \mathbf{x}_j\|}{\sigma_x})^2\right)$, $\mathbf{x}_i$ and $\mathbf{x}_j$ are input points, γ_i and γ_j are leverage parameters of the i-th and j-th inputs, respectively, and $\theta = \{\sigma_f^2, \sigma_x^2\}$ are hyperparameters of the Gaussian process. For test (unseen) inputs, γ will be set to be 1.

Proposition 1: The proposed leveraged kernel function (6) is a positive semi-definite function.

Proof: Let us begin the proof with the basic properties of generating new kernel functions from existing valid kernel functions. Here, we assume that a valid kernel function satisfies the PSD property. Then the sum and product of two valid kernel functions are also valid [1].

One interesting property of a kernel function is that if $k(\mathbf{x_1}, \mathbf{x_1'})$ and $k(\mathbf{x_2}, \mathbf{x_2'})$ are valid over different spaces $\mathcal{X}_1$ and $\mathcal{X}_2$, then the tensor product $k(\mathbf{x}, \mathbf{x}') = k(\mathbf{x_1}, \mathbf{x_1'})k(\mathbf{x_2}, \mathbf{x_2'})$ is also a valid kernel function defined on the product space $\mathcal{X}_1 \times \mathcal{X}_2$. These properties can easily be verified using the definition of the PSD (see Chapter 4 in [1] for details).

Using these properties, we can decompose the kernel function (6) into two parts. The first term is

$$k(\gamma_i, \gamma_j) = (1 - |\gamma_i - \gamma_j|) \tag{7}$$

and the second term is

$$k_{SE}(\mathbf{x}_i, \mathbf{x}_j) = \sigma_f^2 \exp\left(-\frac{1}{2}\left(\frac{\|\mathbf{x}_i - \mathbf{x}_j\|}{\sigma_x}\right)^2\right). \tag{8}$$

The proposed leveraged kernel function (6) is a tensor product of two kernel functions, (7) and (8). As (8) is a well-known squared exponential (SE) kernel which satisfies the PSD, proving (7) is a PSD function with respect to every γ_i and γ_j in $[-1, 1]$ will complete the proof. Note that γ indicates the *leverage* and has values between -1 and 1.

Theorem 3 states that showing the non-negativeness of a Fourier transform of a stationary kernel, i.e. $k(x, x') = k(x - x')$, is equivalent to showing the positive semi-definiteness. Thus, showing the Fourier transform of

$$k(t = \gamma_i - \gamma_j) = 1 - |t| \tag{9}$$

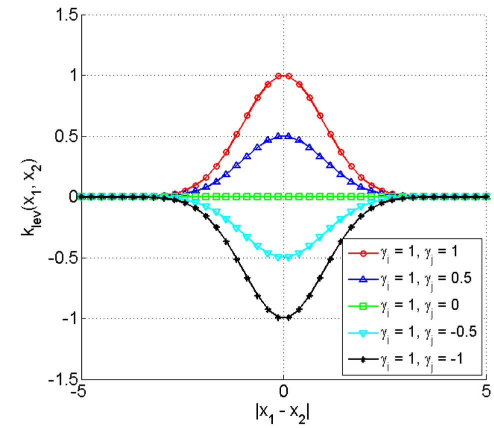


Fig. 1: The proposed leveraged kernel function with different values of γ_i and γ_j.

is non-negative will complete the proof. Since the domain of (9) is $[-2, 2]$, without loss of generality, we can extend (9) to be a periodic function with period 4 outside the domain, i.e., $k(t + 4) = k(t)$.

As the extended function of (9) is a periodic even function, we can express (9) as a Fourier series as follows:

$$\begin{aligned} c_n &= \frac{1}{4}\int_{-2}^{2} k(t) \exp\left(\frac{-i\pi n t}{2}\right) dt \\ &= 2\frac{1 - (-1)^n}{\pi^2 n^2} \\ &= \begin{cases} 0 & \text{for even } n, \\ 4/(n^2\pi^2) & \text{for odd } n. \end{cases} \end{aligned} \tag{10}$$

As the Fourier coefficient (10) is non-negative for all integers, by the Theorem 3, the kernel (9) is positive semi-definite, which completes the proof. ■

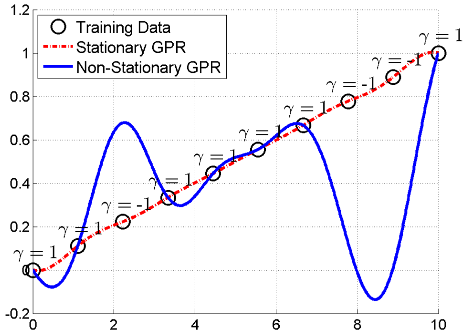
The shape of the proposed leveraged kernel function is shown in Figure 1. We can see that between positive examples (and negative examples), the kernel function works as an ordinary squared-exponential kernel function. However, between positive and negative data, the correlation decreases as the distance between inputs decreases.

As we assume that the *leverage* of unseen data is 1, the positive training data will work as the ordinary training data. However, as being close to the data with a negative *leverage* will lower the correlation, the resulting regressor will be far from such data.

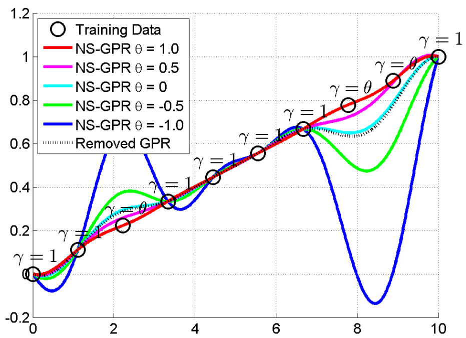
B. Leveraged Non-Stationary Gaussian Process Regression

The non-stationary Gaussian process regression results using the proposed leveraged kernel function are shown in Figure 2(a). The leverages, γ_i, are shown at the top of each training data. As illustrated in Section III-A, the training data with $\gamma = 1$ correspond with the ordinary positive training data and the training data with $\gamma = -1$ work as negative training data. In particular, for those with $\gamma = 0$, such data will have no effect on the resulting regressor.

Figure 2(a) shows regression results of stationary GPR using a squared exponential (SE) kernel and non-stationary



(a)



(b)

Fig. 2: (a) Ordinary Gaussian process regression using a squared exponential (SE) kernel function. (b) Gaussian process regression using the proposed leveraged kernel function.

GPR using the proposed leveraged kernel (6). In particular, for non-stationary GPR, third, eighth, and ninth data points work as negative data ($\gamma = -1$) and the rest of the data work as positive data ($\gamma = 1$). As shown in Figure 2(a), the non-stationary Gaussian process regression tends to *anchor* to the positive data and *drift away* from the negative data.

Figure 2(b) shows how the non-stationary GPR varies with different leverage parameters γ. The removed GPR is the regression result using only the positive training data, i.e., $D = \{(x,y)_i | i = 1,2,4,5,6,7,10\}$. We can see that non-stationary GPR with $\theta = 0$ and removed GPR have similar results as the training data with $\gamma = 0$ have no effect to the regressor in the proposed non-stationary GPR.

IV. Simulation

In this section, we applied the proposed regression framework to a real-time motion control problem using an autoregressive Gaussian process motion controller (AR-GPMC) [3]. In particular, we validate our assumption that incorporating both positive and negative training data will result in a more effective control when given the same number of training data.

A. Gaussian Process Motion Controller

In our previous work [3], a real-time motion controller for dynamic environments is proposed. To be specific, an autoregressive Gaussian process motion controller (AR-GPMC) was proposed to make a safe control in a dynamic environment. An AR-GPMC is a mapping $F_u : \mathcal{T} \to \mathcal{U}$ from a trajectory space $\mathcal{T} \subseteq \mathbb{R}^{2p}$, where p is the length of an input trajectory of a moving obstacle, to the control space $\mathcal{U} \subseteq \mathbb{R}^2$ and Gaussian process regression is used to model the mapping. In other words, an AR-GPMC can control a robot by avoiding an incoming dynamic obstacle given recent p consecutive positions.

First, we used the following squared exponential (SE) kernel for an AR-GPMC:

$$\begin{aligned} k_{SE}(\tau,\tau') &= \sigma_f^2 \exp\left(-\frac{1}{2}\sum_{i=1}^{p}(\tau_i - \tau_i')^T \Sigma_{xy}(\tau_i - \tau_i')\right) \\ &+ \sigma_w^2 \delta_{\tau,\tau'}, \end{aligned} \tag{11}$$

where $\tau \in \mathcal{T}$ is a trajectory of an obstacle, p is the length of the trajectory τ, $\theta = (\sigma_f^2, \Sigma_{xy}, \sigma_w^2)$ are hyperparameters, and $\delta_{\tau,\tau'} = 1$ if $\tau = \tau'$ and zero otherwise.

Then, the motion control is generated by an AR-GPMC as follows:

$$u_\star = k_{SE}(\tau_\star, \tau_{1:N_u})^T \Gamma_u, \tag{12}$$

where

$$\Gamma_u = (k_{SE}(\tau_{1:N_u}, \tau_{1:N_u}))^{-1} u_{1:N_u}, \tag{13}$$

$\tau_\star \in \mathcal{T}$ is the currently observed trajectory, N_u is the number of training data, $\tau_{1:N_u}$ are trajectories in the training data, and $u_{1:N_u}$ are the corresponding controls in the training data.

The proposed non-stationary Gaussian process regression using the leveraged kernel function (6) can be used effectively in this regard as it can incorporate both positive and negative examples. It can be interpreted as training a motion controller using both *what to do* and *what not to do*. In the case of training a motion controller that avoids incoming dynamic obstacles, *what to do* corresponds to the control that dodges approaching obstacles and *what not to do* corresponds to the control that runs into obstacles.

Both positive and negative training data are obtained using sampling-based finite-time horizon control. For example, the positive training data try to reach the goal position while maintaining the minimum distance between the robot and obstacles and the negative training data try to collide with the closest obstacle. Once the training data are collected, we used the determinantal point process (DPP) [9] to select a fixed number of diverse training data.

B. Setup

To make a realistic setup, we assume that a robot can only obtain information from its egocentric view. The field of view (FOV) is set to 120° and this is due to the fact that Microsoft Kinect sensor used in our experiments has 57° FOV and we assume that we use two Kinects for a larger FOV. The number of pedestrians varies from one to six to validate the safety of the proposed method under different conditions. For each setting, 100 independent runs are performed and the goal of the robot is to reach the goal position without any collision.

In order to validate the efficiency of using the non-stationary Gaussian process in motion control, we performed simulations under four different scenarios: 150 positive training data, 300 positive training data, 100 positive and 50 negative training data, and 200 positive and 100 negative training data. When there is no incoming pedestrian, the robot tries to reach the goal using pure pursuit (PP) low-level controller [10].

C. Results

Two criteria have been used to evaluate the performance of the proposed algorithm: the average collision ratio and the total elapsed time to the goal location. The overall results are shown in Figure 3.

Figure 3(a) and 3(b) show the average collision rates and the average minimum distances to the robot from 100 independent runs for different numbers of obstacles. We set the distance threshold for collision to $50\,cm$ which is the minimum detectable distance of the Kinect sensor. As expected, the non-stationary Gaussian process motion controller that uses both positive and negative training data has outperformed the Gaussian process motion controller that uses the positive training data only in terms of the average collision ratio given the same number of training data. The collision ratio of the controller using 100 positive and 50 negative training data is even better than that of using 300 positive training data. The elapsed time of four different scenarios is shown in Figure 3(c) and we can notice that it takes more time to reach the goal when using the non-stationary Gaussian process motion controller as the robot tries to avoid approaching objects by moving backward.

V. EXPERIMENTS

A. Setup

We have conducted field experiments using a Pioneer 3DX differential drive mobile robot with two Microsoft Kinect cameras mounted on top of the robot as shown in Figure 5. All programs are written in MATLAB including mex-compiled ARIA package for controlling a Pioneer mobile robot. The positions of pedestrians are detected using the skeleton grab API of two Kinect cameras which run approximately at $20\,Hz$. The whole system including the pedestrian tracking and the motion control algorithm runs at about $5\,Hz$ on a $2.1\,GHz$ notebook. Eight sonar sensors equipped in the Pioneer platform are used to avoid an immediate collision. This exception routine using sonar sensors is activated if the minimum distance between the robot and pedestrians are less that $50\,cm$ as a Kinect camera cannot detect an object closer than $50\,cm$.

B. Results

We have performed experiments to demonstrate two objectives. The first objective is to make a quantitative experimental comparison between two motion controllers: the non-stationary Gaussian process motion controller (NS-GPMC) trained with both positive and negative training data and the stationary Gaussian process motion controller (GPMC) trained solely with positive training data. The other objective is to show that the proposed motion controller (NS-GPMC) can control a robot safely in a crowded dynamic environment.

TABLE I: Experimental results.

	NS-GPMC	GPMC
Collision rate	$\mathbf{5.0}\,\%$	$15.8\,\%$
Average minimum distance	$\mathbf{1016.2}$ mm	856.1 mm
Average backward time	2.15 sec	1.74 sec

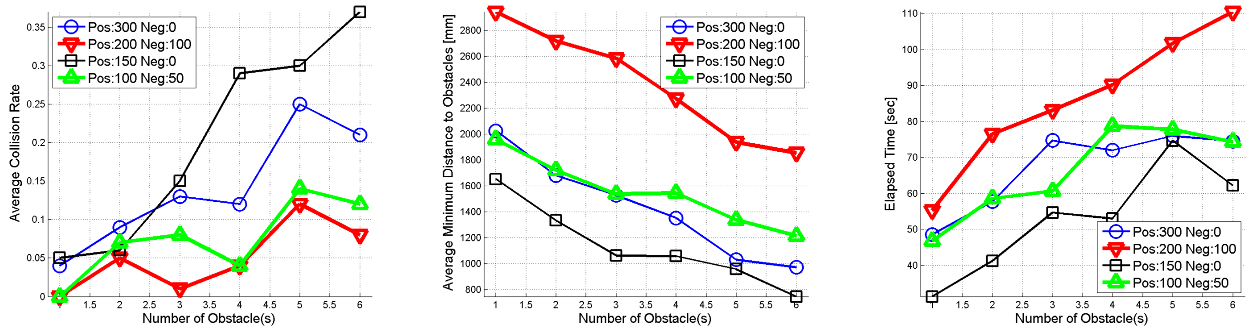
First, we have evaluated the performance of the NS-GPMC and compared to the GPMC. In particular, we used a set with 200 negative and 100 positive data for the NS-GPMC and 300 positive data for the GPMC. In each scenario, 20 experiments are done, and the objective is to reach the goal position, which is at $5\,m$ from the starting position, while avoiding collisions with an incoming pedestrian who follows a predefined path with the same velocity for repeatability. The minimum distance between the robot and the pedestrian is computed using the eight sonar sensors on the Pioneer 3DX mobile robot and the threshold distance for deciding a collision was set to $50\,cm$. As shown in Table I, the NS-GPMC achieves a collision rate of $5.0\,\%$, which is $68\,\%$ lower than GPMC's $15.8\,\%$, while the average minimum distance is $1016.2\,mm$, which is $18.7\,\%$ higher than GPMC's $856.1\,mm$. Furthermore, the average time of a robot moving backward is computed for both methods and it shows that the NS-GPMC is more likely to move backward to avoid an approaching pedestrian.

It is worthwhile to note that all the outputs (controls) of the negative training data have positive velocities, i.e., going straight. However, when these negative training data are applied to the proposed NS-GPMC, the resulting motion is likely to go backward which goes along with the notion of *what not to do*.

We have also tested the proposed NS-GPMC in an L-shaped corridor in a lobby, a basement, and a crowded school cafeteria. Figure 4 shows some snapshots from the experiment in the school cafeteria and the NS-GPMC had successfully navigated the robot in this crowded dynamic environment.

VI. CONCLUSION

In this paper, we have proposed a novel regression framework that can incorporate both positive and negative training data into a single regression framework. In detail, a leveraged kernel function for non-stationary Gaussian process regression is proposed. The leveraged kernel function can not only vary the correlation between two inputs in the positive direction but also to the negative direction using the leverage parameter γ. By using this property, the resulting non-stationary Gaussian process regression can anchor the regressor to the positive data while avoiding the negative data. In addition, we have proven the proposed kernel function is positive semi-definite using Bochner's theorem and characterized its mathematical interpretations. The leveraged non-stationary Gaussian process regression is applied to the motion control problem, in which a controller learns *what*



(a) (b) (c)

Fig. 3: Simulation results. Thick solid lines indicate results using both positive and negative data. (a) Average collision rates of different scenarios for different numbers of obstacles. (b) Average minimum distances to obstacles. (c) Elapsed times to the goal.

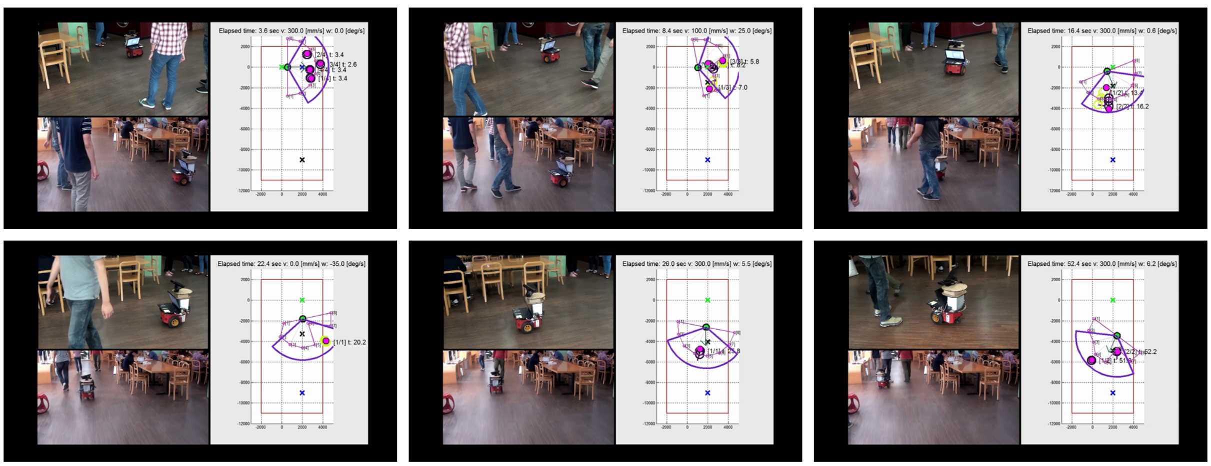


Fig. 4: Snapshots from the school cafeteria experiment.

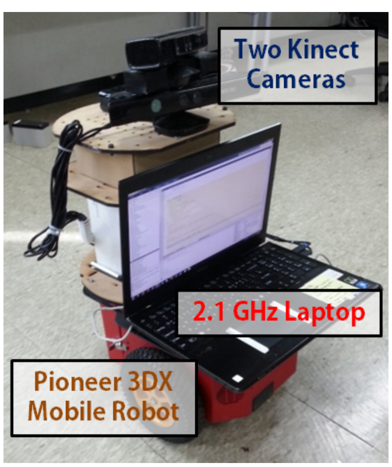


Fig. 5: A robot platform used in experiments.

to do from the positive data and *what not to do* from the negative data. The results show that the controller using the proposed leveraged kernel function based on both positive and negative data outperforms the controller using positive data only when training sets of the same size are used.

REFERENCES

[1] C. Rasmussen and C. Williams, *Gaussian processes for machine learning*. MIT press Cambridge, MA, 2006, vol. 1.

[2] B. Scholkopf and A. J. Smola, *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press Cambridge, MA, 2001.

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

[4] M. Deisenroth, C. Rasmussen, and J. Peters, “Gaussian process dynamic programming,” *Neurocomputing*, vol. 72, no. 7, pp. 1508–1524, 2009.

[5] G. Wachman, R. Khardon, P. Protopapas, and C. R. Alcock, “Kernels for periodic time series arising in astronomy,” in *Machine Learning and Knowledge Discovery in Databases*. Springer, 2009, pp. 489–505.

[6] C. Paciorek and M. Schervish, “Nonstationary covariance functions for Gaussian process regression,” in *Proc. of the Advances in Neural Information Processing Systems*, vol. 16, 2004, pp. 273–280.

[7] D. Higdon, J. Swall, and J. Kern, “Non-stationary spatial modeling,” *Bayesian statistics*, vol. 6, no. 1, pp. 761–768, 1999.

[8] S. Bochner, *Harmonic analysis and the theory of probability*. Courier Dover Publications, 2012.

[9] A. Kulesza and B. Taskar, “k-dpps: Fixed-size determinantal point processes,” in *Proc. of the International Conference of Machine Learning (ICML)*, 2011, pp. 1193–1200.

[10] O. Amidi and C. E. Thorpe, “Integrated mobile robot control,” Robotics Institute, Carnegie Mellon University, Tech. Rep. Robotics Institute, Carnegie Mellon University, 1990.