2013 IEEE/RSJ International Conference on
Intelligent Robots and Systems (IROS)
November 3-7, 2013. Tokyo, Japan

Point Cloud based Dynamical System Modulation for Reactive Avoidance of Convex and Concave Obstacles

Matteo Saveriano and Dongheui Lee

Abstract— **The ability of the robot to avoid undesired collisions with humans and objects in its workspace is of importance in the field of human-robot interaction. In this paper, we propose an algorithm which allows the robot to avoid obstacles and to reach the assigned goal as long as the goal does not lie within obstacles. For this purpose, dynamical system modulation approach is adopted which ensures the avoidance of convex and concave obstacles. A modulation matrix can be calculated directly from the point cloud data of obstacles in the scene, without the need of analytical representation of the obstacles. This matrix modulates a generic first order dynamical system, used to generate the goal. In this way we guarantee the obstacles avoidance and the reaching of the goal. The effectiveness of the proposed approach is validated with numerical simulations and experiments on a 7 DOF KUKA light weight arm.**

I. Introduction

When a human and a robot are in a close cooperation, the robot is required to adapt quickly to various situations and to eventual external disturbances ensuring the operator safety. A quick adaptation means that the robot has to react to several kind of external perturbations in real-time. Examples of these external disturbances are: changes in the goal position to reach, presence of unknown obstacles, accidental contacts with the human or with other objects.

A feasible solution for reacting in real-time to the external perturbations consists in representing the task as a dynamical system (DS). A DS is robust against perturbations, and it ensures the convergence to the goal [1][2]. In order to react to the presence of unknown obstacles and humans, the robot has to modify its motion quickly to avoid collisions. After the avoidance, it is desirable that the robot fulfills the assigned task as long as possible.

The methods for generating collisions free paths can be divided into two categories: path planning approach and reactive motion generation approach. The former is complex global approach which is able to find the shortest collision free path even in very complex scenarios with multi degree-of-freedom robots [3]. Despite the possibility to parallelise the algorithms in order to reduce the computation time [4], at present the computation time is still too large to apply this algorithm on-line.

The latter includes local algorithms which change the robot path in real-time. A widely used approach is based on an artificial potential field [5]. The idea is to assign an attractive force to the goal and to shape the obstacles as

Authors are with Fakultät für Elektrotechnik und Informationstechnik, Technische Universität München, Munich, Germany matteo.saveriano@tum.de, dhlee@tum.de.

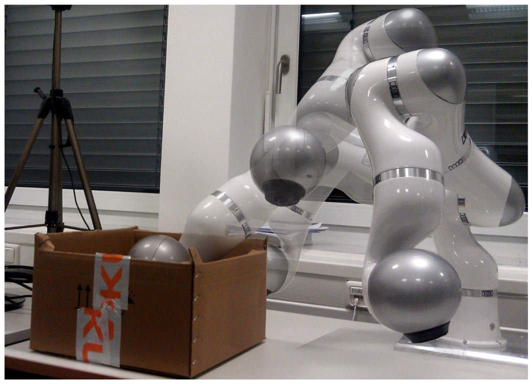


Fig. 1. The KUKA Light-Weight-Robot IV+ goes in and out of a box avoiding collisions. This task is useful, for example, to ask the robot to take something in the box.

repulsive forces, so as to reach the target avoiding obstacles. The potential field technique is applied in [6], where an algorithm to extract information about the location of the obstacles directly from the image plane of an RGB-D sensor is proposed. One drawback of the potential field approach is that the motion can stop in a local minimum even if a collision-free path to the goal exists.

A solution to skip the local minima is proposed in [7] by combining the benefits of the path planning algorithms with the velocity of the reactive techniques. In this method, the initial elastic band is computed off-line using a path planning algorithm, which results in a collision free path. In the presence of obstacle, the band is deformed by applying repulsive forces. However, if the path being executed gets infeasible because of the obstacles coming into its way, the reshaping method cannot be applied any more, and an off-line replanning step is needed [8].

Other researchers propose to avoid local minima by modifying the dynamics of a particular system of differential equations. For example in [9][10] an additive term is applied to a discrete Dynamic Movement Primitive (DMP) [2] in order to deform the trajectory and avoid a point obstacle. The global stability of the modified system is proved with static obstacles using the *Lyapunov theorem*. In [11] a potential field is applied to a second order system with varying stiffness that generates a smooth collision-free path. A combination of potential fields and circular fields is proposed in [12]. Several experiments show the good convergence properties to the goal of this approach, also in very complex scenarios. The mentioned approaches work only with a specific dynamical system, reducing the possibility of encoding many different tasks, such as periodic motions.

A technique to modulate a generic first order DS is proposed in [13]. Given the analytical representations of the obstacles surface, a modulation matrix, which locally deforms the original system, is computed. This approach can be applied on a variety of DS (both stable and unstable) and it guarantees the obstacles impenetrability and it does not modify the equilibria of the modulated system. The modulation technique in [13] is a valid approach to combine safety issues with the robustness of the DS-based motion generation techniques, since it preserves the convergence properties of the modulated system. However, the need of an analytical representation of the surface of the obstacles reduces the effectiveness of this approach in unknown scenarios. Moreover, the algorithm fails in avoiding concave obstacles.

Our proposed approach is based on the dynamical system modulation for obstacle avoidance in [13]. In contrast to the original algorithm, our contributions is twofold. First, an analytical representation of the obstacles is no longer prerequisite. Instead, we use the Euclidean distance of a point from the point clouds of the objects[1]. This gives us the possibility to work in real-time. Second, impenetrability of concave obstacles as well as convex obstacles are achieved in the proposed approach. This property allows the robot larger accessible areas, for example free space in an opened box (Fig. 1). The proposed modulation guarantees the impenetrability of the obstacles without changing the modulated DS equilibrium points. The effectiveness of our approach is proved with simulations and experiments on a KUKA LWR IV+. The proposed method is compared with the technique in [13].

While the original dynamical system modulation method in [13] is based on the analytical representation of the object, which has continuous first order partial derivatives, our proposed algorithm is based on the representation of the object as a point cloud. Thereafter, in this paper we refer the one in [13] as *Continuous Modulation (CM)* and our approach as *Discrete Modulation (DM)*.

The rest of the paper is organised as follows. Section II describes the Continuous Modulation and its main features. In Section III we propose a new point cloud based modulation approach to avoid any geometrically shaped obstacles. Section IV presents the simulation and experimental results. Section V states the conclusions and the future works.

II. Dynamical Systems Modulation Algorithm

This section briefly describes the CM algorithm in [13]. The modulation algorithm is based on the assumption that the path to follow is generated by a first order dynamical system. This system can be autonomous (time-invariant) or non-autonomous (time-variant). Specifying with the state $p \in \mathbb{R}^n$ of the system, the DS becomes:

$$\dot{p}(t) = \phi(p(t)), \qquad \text{autonomous} \qquad (1)$$

$$\dot{p}(t) = \phi(t, p(t)), \qquad \text{non-autonomous} \qquad (2)$$

where $\phi(\cdot)$ is a continuous function and $\dot{p}$ is the first time derivative of p. To ease notation, we omit hereafter the time dependence of variables using ϕ to refer to both autonomous and non-autonomous systems. Knowing the starting point p_0, the desired trajectory can be calculated by integrating ϕ:

$$p_{i+1} = p_i + \phi\delta t, \quad i = 0, 1, \ldots \qquad (3)$$

where δt is the integration time step[2].

By modulating the DS with a suitable matrix $M(p)$

$$\dot{p} = M(p)\phi \qquad (4)$$

one can avoid obstacles and keep the stability properties of the DS. In the following pages, p denotes the generic point, $\bar{p}$ a point on the object surface, and $\tilde{p} = p - p^c$ a generic point with respect to the center of the object p^c.

A. Modulation Matrix Construction

Assume that only one n-dimensional obstacle is present in the scene. Its surface can be described as the zeros locus of a function[3] $\Phi(\tilde{p}) : \mathbb{R}^n \mapsto \mathbb{R}$. For example $\Phi(\tilde{p}) = \tilde{p}^T\tilde{p} - r^2 = 0$ is a n-dimensional hypersphere of center p^c and radius r. Assume that the function Φ is continuous and differentiable so that it is possible to define everywhere the normal to the surface of an obstacle. The normal vector can be calculated as

$$n(\tilde{p}) = \left[\frac{\partial\Phi(\tilde{p})}{\partial\tilde{p}_1} \; \frac{\partial\Phi(\tilde{p})}{\partial\tilde{p}_2} \cdots \frac{\partial\Phi(\tilde{p})}{\partial\tilde{p}_n}\right]^T. \qquad (5)$$

Moreover, suppose that Φ increases monotonically with $\|\tilde{p}\|$. The value of Φ in a generic point p is a measure of the proximity of this point to the surface [14]. In addition, it holds that: $\Phi(\tilde{p}) < 0$ if $\tilde{p}$ is within the surface, $\Phi(\tilde{p}) = 0$ if $\tilde{p}$ belongs to the surface and $\Phi(\tilde{p}) > 0$ if $\tilde{p}$ is external to the surface.

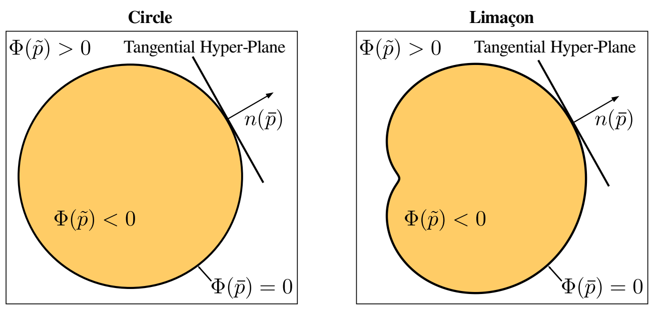


Fig. 2. The tangential hyperplane for the circle and the limaçon.

A tangential hyperplane can be defined at each point on the surface, using the normal vector. By extension, one can

[1]The recent spread of RGB-D sensors makes readily available a representation of the robot workspace as a point cloud. The distance from these points will be used to modulate the DS.

[2]Assuming an integration time step sufficiently small is reasonable to use the *backward Euler* integration method. The use of more sophisticated methods is also possible.

[3]In general, a surface in $\mathbb{R}^n$ can be represented by $\Phi(\tilde{p}) = s, s \in \mathbb{R}$ [14].

define the hyperplane at all points of the space as $\Phi(\tilde{p}) \geq 0$. One particular basis of the tangential hyperplane is

$$v_i^j(\tilde{p}) = \begin{cases} -\dfrac{\partial \Phi(\tilde{p})}{\partial \tilde{p}_{i+1}} & j = 1 \\ \dfrac{\partial \Phi(\tilde{p})}{\partial \tilde{p}_1} & j = i+1 \quad i = 1..n-1, j = 1..n \\ 0 & j \neq 1, j \neq i+1 \end{cases} \tag{6}$$

where v_i^j corresponds to the j-th component of the i-th basis vector. Note that the components of each $v_i(\tilde{p})$ can be directly computed from the components of $n(\tilde{p})$.

The matrix $V(\tilde{p}) = [n(\tilde{p}) \;\; v_1(\tilde{p}) \cdots v_{n-1}(\tilde{p})]$ is an orthonormal basis of the n-dimensional space. The normal vectors and the corresponding tangential hyperplane are shown in Fig. 2, for a circle and a limaçon[4]. The yellow area represents the internal points ($\Phi < 0$), the white one the external points ($\Phi > 0$).

The diagonal matrix $E(\tilde{p})$ is defined as

$$E(\tilde{p}) = \begin{bmatrix} \lambda_1(\tilde{p}) & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \lambda_n(\tilde{p}) \end{bmatrix} \tag{7}$$

where

$$\begin{cases} \lambda_1(\tilde{p}) = 1 - \dfrac{1}{|\Phi(\tilde{p}) + 1|} \\ \lambda_i(\tilde{p}) = 1 + \dfrac{1}{|\Phi(\tilde{p}) + 1|} & i = 2, 3, \ldots n \end{cases} \tag{8}$$

The modulation matrix can be calculated as

$$M(\tilde{p}) = V(\tilde{p}) E(\tilde{p}) V(\tilde{p})^{-1} \ . \tag{9}$$

By modulating (4) with the matrix (9), it is possible to prove that a trajectory $\tilde{p}(t)$ starting from outside an obstacle can never penetrate the convex obstacle[5]: $\Phi(\tilde{p}(0)) \geq 0 \longmapsto \Phi(\tilde{p}(t)) \geq 0, \ \forall t > 0$.

Note that the modulation does not affect the equilibrium points of the modulated DS, saving its stability properties. For the sake of brevity we refer to [13] for the proof of the impenetrability and for the DS stability analysis. Although the modulation technique might generate spurious equilibrium points, a technique to escape this points is discussed in [13].

The modulation consists basically in a local deformation of the DS, that generates collision free paths. The effect of the modulation is maximum at the boundary of the obstacle, and vanishes for points far from it. In fact $\Phi(\tilde{p})$ monotonically increases with $\|\tilde{p}\|$ and the matrix converges to the identity matrix as the distance to the surface increases.

[4]The limaçon is a plane curve obtained when a circle rolls around the outside of a circle of equal radius. It is defined by $(\tilde{p}_1^2 + \tilde{p}_2^2 - b\tilde{p}_1)^2 - a^2(\tilde{p}_1^2 + \tilde{p}_2^2) = 0$, with $b > 0$, $a \geq 0$. If $a \geq 2b$, it is convex. If $b \leq a < 2b$, it is concave. If $a \leq b$ the curve is no more differentiable.

[5]Note that the impenetrability does not depend on the choice made for the base v_i.

B. Extension to Multiple Obstacles

The extension of the presented algorithm to guarantee the avoidance of multiple obstacle is not trivial. Indeed, a combined modulation matrix $\bar{M}$ cannot be calculated by multiplying the modulation matrices of each object, because this does not guarantee the impenetrability. A method to overcome this problem is to weight the contribution given to $\bar{M}$ by each object according to the distance from the robot.

Assumed the presence of K objects in the scene and defined as Φ_k, $k = 1, 2, \ldots K$ their analytical representations (a measure of the distances between the robot position and the K obstacles), these weights are calculated as

$$\omega_k = \begin{cases} 1 & K = 1 \\ \displaystyle\prod_{i=1, i \neq k}^{K} \frac{\Phi_i}{\Phi_k + \Phi_i} & K > 1 \end{cases} \tag{10}$$

where the dependence on p is omitted. The ω_k are continuous positive scalars between zero and one. In addition, on the boundary of the k-th obstacle, it is $\omega_k = 1$ and $\omega_i = 0, \forall i \neq k$. Using these two properties it is possible to demonstrate that the impenetrability and the equilibria of the modulated system are preserved (see [13] for further details).

Given the weights, it is possible to construct the k-th modulation matrix $M_k = V_k E_k V_k^{-1}$. The elements of the matrix E_k are calculated as follows

$$\begin{cases} \lambda_{1,k} = 1 - \dfrac{\omega_k}{|\Phi + 1|^{\frac{1}{\rho}}} \\ \lambda_{i,k} = 1 + \dfrac{\omega_k}{|\Phi + 1|^{\frac{1}{\rho}}} & i = 2, 3, \ldots n \end{cases} \tag{11}$$

Repeating this procedure for each obstacle we obtain k matrices, the product of which will give the combined modulation matrix:

$$\bar{M} = \prod_{i=1, i \neq k}^{K} M_k \ . \tag{12}$$

III. Discrete Modulation Algorithm

The Continuous Modulation requires an analytical description of the obstacle, as a continuous and differentiable function $\Phi(p)$. In a realistic scenario, however, it is not always easy to get an analytical representation of arbitrary objects from raw sensory data such as 3D point cloud from a RGB-D sensor. [13] mentioned how to get an analytical expression from the point cloud: either by assuming a convex hull using [15] or by approximating each object with an ellipsoid for a real-time purpose.

On the other hand, our proposed solution consists in approximating $\Phi(p)$ with the distance between two points. This approach can work in real-time and find a feasible collision-free path also in situations where the CM fails. This algorithm can be generalised for one or multiple obstacles.

A. Distance-based Modulation

Let us consider a single object and a set P_g, $g = 1, \ldots, G$ of points $\mathbb{R}^n$ belonging to the obstacle surface. The function $\Phi(p)$ can be approximated with the Euclidean distance of p

from the surface points. In fact, the distance is zero on the surface and grows with continuity moving away from it[6]. So, the approximated representation of the object surface $\Phi_a(p)$ is calculated as

$$D(p) = min\left(\sum_{i=1}^{n}(P_{1,i}-p_i)^2,\dots,\sum_{i=1}^{n}(P_{G,i}-p_i)^2\right) \tag{13}$$

$$\Phi_a(p) = \sqrt{D(p)} \tag{14}$$

where $P_{g,i}$ denotes the i-th component of the g-th point and p_i the i-th component of the current status of the DS. Hereafter we indicate $\Phi_a(p)$ only with Φ.

To compute the modulation matrix (9), it is necessary to estimate the normal vector at the point of minimum distance. This estimation is carried out using a parallel implementation of the algorithm in [16], which allows to quickly reconstruct a surface from an unordered point cloud. For each point P_g a weighted least squares plane π_g is calculated using points in a neighbourhood of P_g. Then, the normal at P_g is chosen equal to the normal to the plane π_g. This algorithm is robust to noise, which is beneficial when using the RGB-D cameras. Compared to other approaches, it does not require that the surface is smooth nor a certain density of points.

Since the density of the point cloud is not fixed, the direction of the normal vector may vary significantly also in close points. In order to reduce discontinuities in the modulated path, we calculate the normal $n(P_g)$ at a point P_g by using a weighted average on the L points[7] closer to P_g:

$$\begin{aligned} n(P_g) &= cn(P_g) + (1-c)n_{av}(P_g) \\ &= cn(P_g) + (1-c)\frac{1}{L}\sum_{i=1,i\neq k}^{L} n(P_i) \end{aligned} \tag{15}$$

where the scalar $0 \leq c \leq 1$. To ensure the impenetrability, it must be $c = 1$ if $\Phi(p) = 0$. In addition, c should be small when the robot is far from the obstacles, and it should continuously increase as $\Phi(p)$ decreases. For this reason c is calculated as:

$$c = \frac{1}{|\Phi(p)+1|^{\beta}} \tag{16}$$

where the scalar $\beta \geq 0$ is a tunable parameter. For $\beta = 0$, c becomes 1 and the contribution of $n_{av}(P_g)$ in (15) is neglected. Fig. 3 shows that greater value of β allows to obtain smoother trajectories, because the contribution given to the normal by $n_{av}(P_g)$ is dominant till the proximity to the surface.

The value of β mainly depends on the number of points used to represent the obstacle surface. Few points leads bigger values of β, since the normals can significantly change between close points. In our experiments, $\beta = 10$ gives acceptable results.

[6]A more precise approximation is the *signed distance* from the surface, so that would be true $\Phi_a(p) < 0$ if p is inside the surface. This distance can be calculated as $n(P_g)^T(p - P_g)$, where $n(P_g)$ is the unit normal vector at the point P_g. This approach, however, would create problems in the task of going into a box. In fact, a repulsive action would be generated since $\Phi_a(p) < 0$ for the points inside the box. This would keep the robot out of the box. In Section IV it will be also shown that the choice of using the Euclidean distance does not affect at all the obstacle avoidance.

[7]In experiments we use $L = 1\%$ of the number of points.

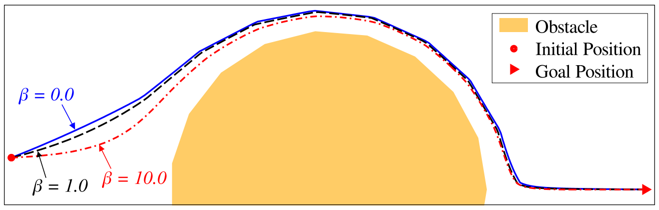


Fig. 3. Trajectories generated calculating the modulation matrix with different values of β.

The described distance-based modulation can be directly applied in a same way, no matter how many obstacles exist in the work space. We simply calculate the distance from the closest point (14), (13) and the normal at this point (15), (16). This means that the number of objects does not affect the performance of our algorithm. On the other hand, the CM requires a step to cluster the point belonging to each obstacle.

Our approximation still guarantees the impenetrability of multiple convex obstacles and does not change the equilibria of the modulated DS. Indeed, assume a globally asymptotically stable DS. So, the velocity vanishes only at the global equilibrium p^*, i.e. $\phi(p^*) = 0$ and $\lim_{t\to\infty}\phi(t,p^*) = 0$, respectively for an autonomous and a non-autonomous DS. When M is full rank, still the velocity vanishes only in p^*, $M(p^*)\phi(p^*) = 0$ and $\lim_{t\to\infty} M(p^*)\phi(t,p^*) = 0$, and p^* remains an equilibrium point.

On the obstacle boundary, M loses one rank and some spurious equilibria can appear. To escape this equilibria, a small perturbation is applied to the state along one of the hyperplane directions $v_1 \dots v_{n-1}$, until the robot is driven out from the basin of attraction of the equilibrium. Since this algorithm is proposed in [13], we refer to that work for further details.

B. *Impenetrability of concave obstacles*

In [13], the impenetrability for convex obstacles is shown by proving that the normal speed at the obstacle surface vanishes:

$$n(\bar{p})^T\dot{\bar{p}} = n(\bar{p})^T M(\bar{p})\phi = 0\ . \tag{17}$$

If (17) holds, the speed in a point $\bar{p}$ on the boundary of the object has non-zero components only on the tangential hyperplane. Since the tangential hyperplane never intersects a convex surface, the impenetrability of a convex obstacle can be immediately assumed[8].

If the object is concave, the tangential hyperplane can intersect the surface, as shown in Fig. 4. Therefore, the point $p = \bar{p} + M(\bar{p})\phi\delta t$, calculated by integrating (4), may be located within the object. To prove the impenetrability, consider the set $I = \{p_I|\Phi(p_I) \geq 0, \forall p_I \in \mathbb{R}^n\}$ that is the

[8]Note that the time discrete implementation of the algorithm can compromise the impenetrability if the integration time step is not sufficiently small. In all experiments, it will be used $\delta t = 1ms$.

set of all the points external or belonging to the surface[9]. A subset of I is defined by

$$J_r(\bar{p}) = \{p_{J_r} | r \geq max(p - \bar{p}), \forall p_{J_r} \in I \subset \mathbb{R}^n\} \quad (18)$$

where $p - \bar{p} = M(\bar{p})\phi\delta t$. By construction, $J_r(\bar{p})$ is an intersection-free neighbourhood of $\bar{p}$.

The impenetrability of a concave object is ensured, if a neighbourhood like (18) exists for each point on the boundary of the obstacle. If this holds, we can conclude that each point p, calculated by integrating (4), is external or belonging to the surface. Therefore, the concave obstacle will be never penetrated.

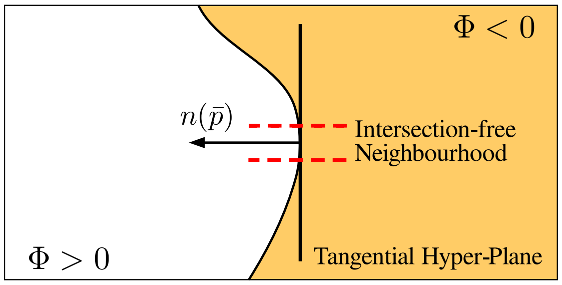


Fig. 4. In a concave object the tangent plane can intersect the surface. If the surface is smooth, a neighbourhood of the point of tangency with no intersections (here is the area between the red dashed lines) can exist.

In order to clarify the concept of impenetrability of a concave obstacle, a simple example is given. Consider a box with one open side, as shown in Fig. 1, and a robot is commanded to move inside the box from its initial position outside of the box. When the robot reaches the goal in the box without hitting the box, we can say that the impenetrability is ensured.

An interesting property of the proposed DM is that we can find a neighbourhood $J_r(p)$, like (18), for each boundary point. The normal at a point p, in fact, is the normal of a weighted least squares plane π, calculated using points in a neighbourhood of p (see Section III-A). Since π is the tangential hyperplane and the neighbourhood used to calculate π is $J_r(p)$, we can guarantee the impenetrability by setting properly the radius r.

C. Customize the Collision-Free Path

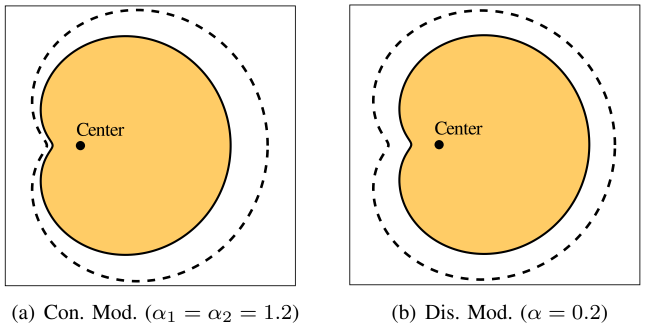
Some parameters were proposed in order to customize the path according to the object and to the robot properties in the CM. Herein, we modify those parameters for our purpose.

First, since the robot end-effector is not a point, it is useful to determine how close to the object the robot can pass. For this purpose we introduce a *safety margin*, a positive scalar $\alpha \in \mathbb{R}$, to calculate the new distance from the obstacle $\Phi(p)'$ as:

$$\Phi(p)' = \Phi(p) - \alpha\ . \quad (19)$$

The trajectories of the system for three values of α can be seen in the Fig. 6(a). The dashed line marks the safe area. The value of α depends on the robot end-effector size. In our experiments on the KUKA LWR we choose $\alpha = 0.08$.

Note that the CM uses a *safety margin* for each dimension $\alpha_i \geq 1$, $i = 1, \dots, n$ to modify the point $\tilde{p} = [\tilde{p}_1/\alpha_1 \dots \tilde{p}_n/\alpha_n]^T$. This allows to set different distances for each dimension. When the object is not symmetric, its center can be close to the boundary. In such a case, this can generate a safety area that is too close to the obstacle surface, as shown in Fig. 5(a). Instead, the modified *safety margin* in (19) generates a safety area in which every point is at distance α from the surface, as in Fig. 5(b).



(a) Con. Mod. ($\alpha_1 = \alpha_2 = 1.2$) (b) Dis. Mod. ($\alpha = 0.2$)

Fig. 5. The safety areas around a limaçon obtained using (a) the CM and (b) the DM *safety margin*.

Changing the magnitude of the modulation can be useful to avoid slow-moving objects, because the robot reacts earlier to the obstacle and it moves away from the trajectory of the object. Thus, a *reactivity* parameter is introduced amending the eigenvalues of the modulation matrix (9) as follows

$$\begin{cases} \lambda_1(p) = 1 - \dfrac{1}{|\Phi(p) + 1|^{\frac{1}{\rho}}} \\ \lambda_i(p) = 1 + \dfrac{1}{|\Phi(p) + 1|^{\frac{1}{\rho}}} & i = 2, 3, \dots n \end{cases} \quad (20)$$

As the *reactivity* parameter ρ increases, the distance at which the object begins to be perceived increases. This means that from that distance the matrix M deviates significantly from the identity matrix. In other words the local deformation of the DS will start earlier if $\rho > 1$, or it will be delayed if $\rho < 1$. The results obtained modifying ρ are shown in Fig. 6(b). In our experiments on the KUKA LWR we choose $\rho = 0.3$, to avoid sudden movements at the beginning of the motion due to the obstacle proximity.

The last desired property is the ability to interrupt the modulation once the robot passes the object. The interruption of the modulation is obtained introducing the boolean variable $m = 0, 1$ and redefining the first eigenvalue of the modulation matrix as

$$\lambda_1(p) = \begin{cases} 1 - \dfrac{1}{|\Phi(p) + 1|^{\frac{1}{\rho}}} & \dot{p}^T\tilde{p} < 0 \text{ or } m = 1 \\ 1 & \dot{p}^T\tilde{p} \geq 0 \text{ and } m = 0 \end{cases} \quad (21)$$

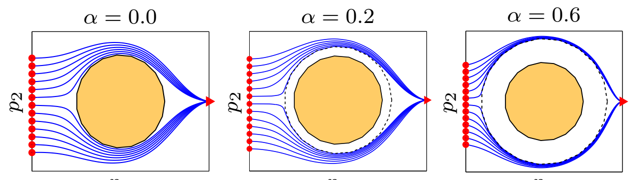
where $\tilde{p}$ is the position with respect to the center of the obstacle[10]. The sign of $\dot{p}^T\tilde{p}$ is used to determine if the robot

[9]In our approximation it is $\Phi > 0$ both for points internal and external to the obstacle surface. Since the sign of Φ does not affect what follows, we still indicate with $\Phi > 0$ the external points.

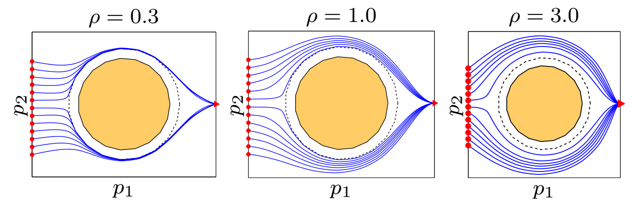
[10]Note that in (21) only $\lambda_1(p) = 1$, since the modulation of the other components is anyway necessary to ensure the continuity of the velocity.

is approaching the object (negative sign) or if it is moving away (positive sign) from it. If the system is continuously modulated ($m = 1$) the path follows the obstacle shape also after passing the obstacle. Fig. 6(c) represents the modulated trajectories, obtained with $m = 1$ and $m = 0$.

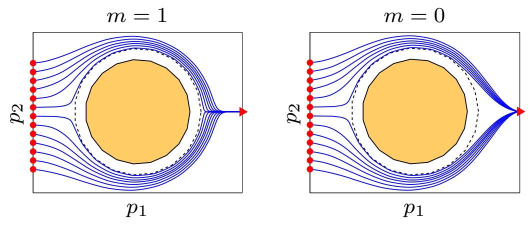
Note that the CM uses the sign of $n(\tilde{p})^T\dot{p}$. The sign of $n(\tilde{p})^T\dot{p}$ can only be used if $n(\tilde{p})^T$ is the normal vector outgoing from a closed surface. Since we rarely deal with closed surfaces due to partial occlusion in the RGB-D sensor, we decided to use (21). The introduced parameters do not affect the impenetrability nor the equilibria of the DS.



(a) Safety Margin



(b) Reactivity



(c) Modulation Interrupt (Off/On)

Fig. 6. Characterizing the path during the obstacle avoidance. The scalars α and ρ are used to modify the size of the safe area and the magnitude of the modulation. The boolean value m is used to interrupt the modulation after passing the obstacle.

The complete DM is summarized in Algorithm 1.

Algorithm 1 Discrete Modulation

```
Given a point cloud representing $K$ obstacles and the
normal vector at each point
  1. Calculate $\Phi$ using (13), (14), (19)
  2. Smooth the relative normal using (15), (16)
  3. Calculate $\lambda_1$ using (21) and $\lambda_i,\ i = 2, \dots n$ using (20)
  4. Calculate $E$ using (7)
  5. $V = [n \quad v_1 \cdots v_{n-1}]$
return $M = VEV^{-1}$
```

This proposed technique is applied in a two-dimensional simulation and the simulation results are shown in Fig. 7. The desired trajectory $p(t)$ is a straight line from the initial position to a global asymptotically stable equilibrium g in the case of no obstacles. This equilibrium is generated using the linear system $\dot{p}(t) = K(g - p(t)),\ K = 10$. For the simulation, the following parameters were used: $\beta = 10$, $\alpha = 0.2$, $\rho = 1$ and $m = 0$. Starting from different initial states, the modulated trajectories always reach the equilibrium point. The algorithm finds a smooth and feasible path, also when only very narrow passages between the obstacles exist, as long as the safety margin is ensured.

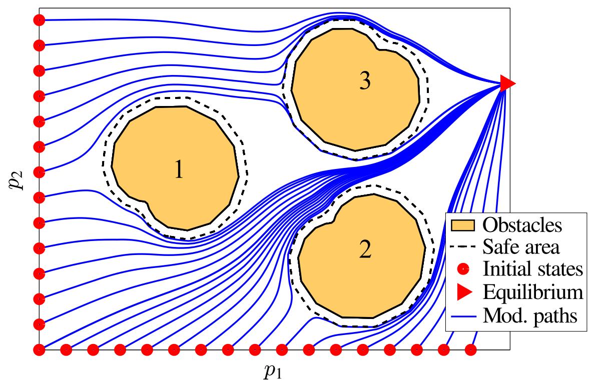


Fig. 7. Modulation with three obstacle in a 2-dimensional space, starting from different position. The original path $p(t)$ is a straight line (from the initial state to the equilibrium g), calculated by integrating $\dot{p}(t) = K(g - p(t)),\ K = 10$.

IV. Experimental Results

A. Implementation Details

The proposed Discrete Modulation has been implemented both in Matlab and C++ in order to test in simulated and real environments. Real experiments are conducted on a KUKA LWR IV+, controlled at 1kHz by an external PC (*Intel I5* quad-core processor) using the *Fast Research Interface* [17].

The scene is monitored using a Microsoft Kinect RGB-D sensor, which gives a 640×480 matrix of 3D points. A *ROS (Robot Operating System*[11]*)* node implements a shader-based filter[12]. This node takes as inputs the current point cloud, the camera frame and the robot geometry, and removes the points on the robot surface and those outside its workspace. The filtered points are used to calculate the modulation matrix. The normal at each point are calculated using the parallel implementation of [16] provided in the *Point Cloud Library (PCL)*[13]. The update of the modulation matrix occurs every 30Hz.

A second ROS node, which runs at 1kHz, generates the desired trajectory modulating a DS, sends the collision free path to the robot and updates its current position. Since the frequencies of the nodes are different, the DS is modulated always using the last calculated matrix, so that the robot is constantly controlled at 1kHz.

This software architecture, consisting of two separated ROS nodes, allows the execution of each node in a separate thread with different scheduling priorities. In this way, the robot control node is not affected from the other operations.

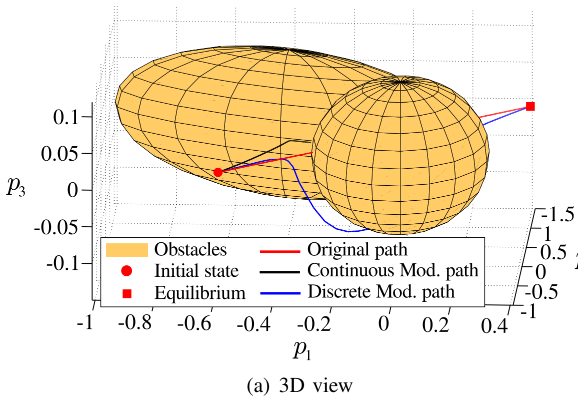
[11]`www.ros.org/wiki`
[12]`github.com/jhu-lcsr-forks/realtime_urdf_filter`
[13]`www.pointclouds.org`

B. Numerical Simulations

In this simulation, we show a comparison between the CM [13] and the proposed DM. In the scene there is a 3-dimensional occluded region which consists of two attached ellipsoids. The original path is calculated by integrating $\dot{p}(t) = K(g - p(t))$, $K = 10$. For the simulation, the following parameters were used: $\beta = 10$, $\alpha = 0.1$, $\rho = 1$ and $m = 0$.

As shown in Fig. 8, the DM find a free path to the goal, while the CM stops at a local minimum. In the CM, we modelled the obstacle as two ellipsoids[14]. Because the two ellipsoids are attached, it can happen that the distances from the two objects are null at the same time, so divisions by zero occur in (10)[15]. It cannot directly deal with concave shaped occlusions. To avoid the problem with the CM it is necessary to calculate an analytical representation of the convex hull, which leads the loss of a lot of accessible free area.

In contrast, the DM does not need an analytical representation of the obstacle. At each iteration, it simply finds the closest point and uses this point to calculate the modulation matrix. So, a smooth and feasible path to the goal is found.



(a) 3D view

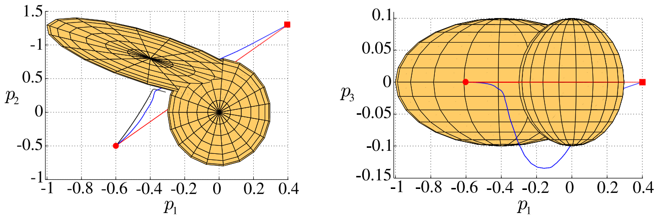
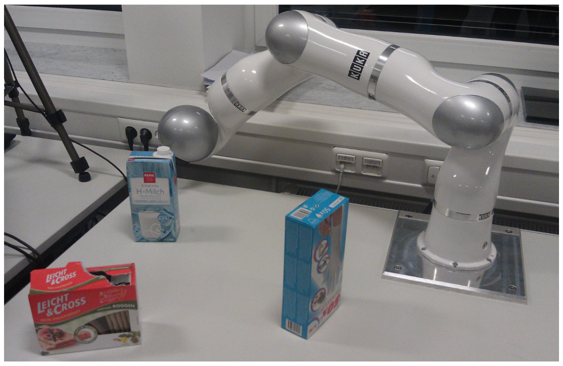

(b) Top view (c) Lateral view

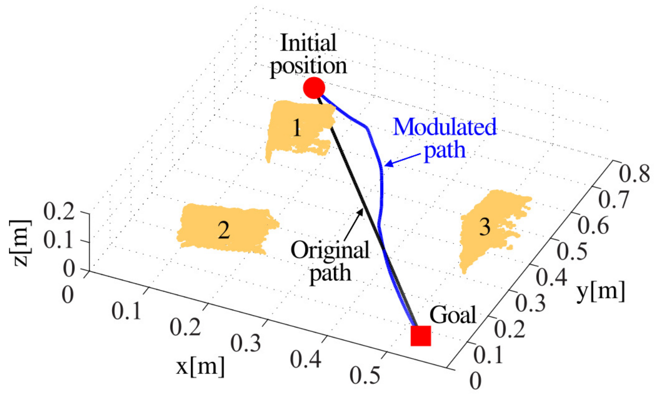
Fig. 8. A comparison between the CM and the DM, with an obstacle made by two attached ellipsoids. The CM fails with connected objects when more then one $\omega_k = 0$ exist.

C. Robot Experiments

Avoiding Multiple Obstacles: In this experiment the robot must avoid three obstacles in the scene, as shown in Fig. 9(a). The original path is generated using $\dot{p}(t) = K(g - p(t))$, $K = 2$. The initial position and the goal g, together with the dimensions of the obstacles, are chosen so that the robot passes through the obstacles and not above them.

Fig. 9(b) represents the results obtained modulating the system with $\beta = 10$, $\alpha = 0.08$, $\rho = 0.3$ and $m = 0$. The input point cloud consists of about 14300 points, and the mean computation time to calculate the minimum distance and to estimate the normal is $0.32ms$. In Fig. 9(b), the distortion of the trajectory due to the modulation is clearly visible since the beginning of the movement, since the robot starts near the obstacle 1. After passing the first obstacle, the modulated trajectory converges to the original one. In the final stage the robot remains close to the original path, managing to pass safely between the objects 2 and 3 (25cm is the distance between them). Here, the contribution given by 2 and 3 to the modulation matrix tends to balance being the obstacles almost equidistant from the original path.


(a) Experiment set-up



(b) 3D view

Fig. 9. Multiple obstacles avoidance. Given the point clouds (yellow points) of the obstacles, a collision-free path to the goal is found using the DM.

Going into and out of a box: In this experiment the robot has to reach two goal positions, one of which is located at the center of a box of size 40cm×35cm×20cm. One side of the box is open. This set-up is depicted in Fig. 1.

Starting from a point outside the box, the robot is guided to the first goal g_1 inside it by the system $\dot{p}(t) = K(g_1 - p(t))$, $K = 2$. Then, starting from g_1 the robot comes out of the box and reaches the second goal g_2 of $\dot{p}(t) = K(g_2 - p(t))$, $K = 2$. A collisions free path is found by modulating this switching linear DS, as represented in Fig. V. The input

[14]Since we cannot find an analytical representation of the obstacle surface.
[15]In general this problem arises every time that more then one $\Phi_k = 0$ at the same time.

point cloud consists of about 32000 points, and the mean computation time to calculate the minimum distance and to estimate the normal is $0.59ms$. These results are obtained with $\beta = 10$, $\alpha = 0.08$, $\rho = 0.3$ and $m = 1$.

In this experiment it is necessary that $m = 1$ in the output step. If it is not, the robot would hit the box because the modulation would be interrupted since $\dot{p}^T \tilde{p} \geq 0$ and, from (21), $\lambda_1 = 1$.

Note that, by using the CM, the task cannot be accomplished. In this case, the box should be approximated using a bounding box, thereby preventing the robot to enter. Even if the box is approximated considering each face as a separate obstacle, there may be division by zero and the algorithm crashes (see Section IV-B for further details). Our approach does not have this problem, because it modulates the system only considering the closest point at each iteration.

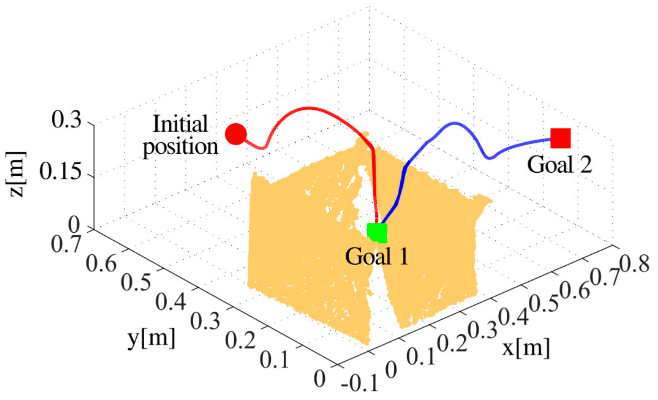


Fig. 10. The task of going into and out of a box. Given the point cloud (yellow points) of the box and two goal positions, the robot is driven into (red line) and out (blue line) of a box modulating a switching linear DS.

V. Conclusion and Future Work

In this paper we presented a real-time collision avoidance technique that guarantees the convergence to the goal. The novelty of our approach lies in the calculation of the modulation matrix directly from a point cloud. This makes the resulting algorithm faster than other approaches which requires an analytical representation of the obstacle surface. It is shown that this algorithm works with convex and concave objects without being stuck into local minima. This approach allows the robot to access all free area outside of the safety area around the object and leads to relatively short collision-free path to the goal. We implemented and evaluated the point cloud based dynamical system modulation approach on a 7 DOF KUKA LWR with comparison of the original dynamical system modulation approach. The experimental results show the clear advantages which can handle connected multiple obstacles and concave obstacles.

In the future, we will focus on extend the Discrete Modulation to a more realistic scenario, in which the robot interacts with humans and several moving obstacles. This approach will be extended to avoid collision with the whole robot by projecting the movements in the robot null space.

Acknowledgements

This work has been partially supported by the DFG excellence initiative research cluster ”Cognition for Technical System CoTeSys” and the European Community within the FP7 ICT-287513 SAPHARI project.

References

[1] S. M. Khansari-Zadeh and A. Billard, “Learning stable non-linear dynamical systems with gaussian mixture models,” *Transaction on Robotics*, vol. 27, no. 5, pp. 943–957, 2011.

[2] A. Ijspeert, J. Nakanishi, P. Pastor, H. Hoffmann, and S. Schaal, “Dynamical Movement Primitives: learning attractor models for motor behaviors,” *Neural Computation*, no. 25, pp. 328–373, 2013.

[3] H. Choset, K. M. Lynch, S. Hutchinson, G. A. Kantor, W. Burgard, L. E. Kavraki, and S. Thrun, *Principles of Robot Motion: Theory, Algorithms, and Implementations*. MIT Press, 2005.

[4] A. Vazquez-Otero, J. Faigl, and A. P. Munuzuri, “Path planning based on reaction-diffusion process,” in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 896–901.

[5] O. Khatib, “Real-time obstacle avoidance for manipulators and mobile robots,” *International Journal of Robotics Research*, vol. 5, no. 1, pp. 90–98, 1986.

[6] F. Flacco, T. Kröger, A. De Luca, and O. Khatib, “A depth space approach to human-robot collision avoidance,” in *Proc. of the IEEE International Conference on Robotics and Automation*, 2012, pp. 338–345.

[7] O. Brock and O. Khatib, “Elastic strips: A framework for motion generation in human environments,” *The International Journal of Robotics Research*, vol. 21, no. 12, pp. 1031–1052, 2002.

[8] E. Yoshida and F. Kanehiro, “Reactive robot motion using path replanning and deformation,” in *Proc. of the IEEE International Conference on Robotics and Automation*, 2011, pp. 5456 – 5462.

[9] D.-H. Park, H. Hoffmann, P. Pastor, and S. Schaal, “Movement reproduction and obstacle avoidance with dynamic movement primitives and potential fields,” in *Proc. of the IEEE/RAS International Conference on Humanoid Robotics*, 2008, pp. 91–98.

[10] H. Hoffmann, P. Pastor, D.-H. Park, and S. Schaal, “Biologically-inspired dynamical systems for movement generation: automatic real-time goal adaptation and obstacle avoidance,” in *Proc. of the IEEE international conference on Robotics and Automation*, 2009, pp. 1534–1539.

[11] S. Haddadin, H. Urbanek, S. Parusel, D. Burschka, J. Roßmann, A. Albu-Schäffer, and G. Hirzinger, “Real-time reactive motion generation based on variable attractor dynamics and shaped velocities,” in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2010, pp. 3109–3116.

[12] S. Haddadin, S. Belder, and A. Albu-Schäffer, “Dynamic motion planning for robots in partially unknown environments,” in *IFAC World Congress*, 2011, pp. 6842–6850.

[13] S. M. Khansari-Zadeh and A. Billard, “A dynamical system approach to realtime obstacle avoidance,” *Autonomous Robots*, vol. 32, no. 4, pp. 433–454, 2012.

[14] J. Bloomenthal and B. Wyvill, Eds., *Introduction to Implicit Surfaces*. Morgan Kaufmann Publishers Inc., 1997.

[15] M. Benallegue, A. Escande, S. Miossec, and A. Kheddar, “Fast C1 proximity queries using support mapping of sphere-torus-patches bounding volumes,” in *Proc. of the IEEE International Conference on Robotics and Automation*, 2009, pp. 483–488.

[16] Z. C. Marton, R. B. Rusu, and M. Beetz, “On fast surface reconstruction methods for large and noisy datasets,” in *Proc. of the IEEE International Conference on Robotics and Automation*, 2009, pp. 3218–3223.

[17] G. Schreiber, A. Stemmer, and R. Bischoff, “The fast research interface for the kuka lightweight robot,” in *Proc. of the IEEE ICRA Workshop on Innovative Robot Control Architectures for Demanding (Research) Applications - How to Modify and Enhance Commercial Controllers*, 2010, pp. 15–21.