Auton Robot (2012) 32:433–454
DOI 10.1007/s10514-012-9287-y

A dynamical system approach to realtime obstacle avoidance

Seyed Mohammad Khansari-Zadeh · Aude Billard

Received: 1 May 2011 / Accepted: 8 March 2012 / Published online: 23 March 2012

Abstract This paper presents a novel approach to real-time obstacle avoidance based on Dynamical Systems (DS) that ensures impenetrability of multiple convex shaped objects. The proposed method can be applied to perform obstacle avoidance in Cartesian and Joint spaces and using both autonomous and non-autonomous DS-based controllers. Obstacle avoidance proceeds by modulating the original dynamics of the controller. The modulation is parameterizable and allows to determine a safety margin and to increase the robot's reactiveness in the face of uncertainty in the localization of the obstacle. The method is validated in simulation on different types of DS including locally and globally asymptotically stable DS, autonomous and non-autonomous DS, limit cycles, and unstable DS. Further, we verify it in several robot experiments on the 7 degrees of freedom Barrett WAM arm.

Keywords Realtime obstacle avoidance · Nonlinear dynamical system · Harmonic potential function · Robot manipulator

1 Introduction

In our quest to develop robots that react to arbitrary forms of perturbations, we seek methods by which this reactivity will be effortless and will unfold naturally from the control law. Imagine you are being served tea by a robot. As the robot is about to pour the boiling liquid in the cup you are holding, you sneeze. As a result of your sudden hiccup, the cup is displaced and your hand is now in the way of the robot in place of the cup. Surely, you wish the robot would be able to react swiftly, so as to redirect its motion to the cup while avoiding your hand. These are examples of fast perturbations that require a reactivity of the order of the second. These encompass a wide variety of perturbations dealt with by robotics such as: when an obstacle suddenly appears in the robot's path, when the target moves, or when the robot is pushed away from its trajectory while in motion. In these situations, there is no time to re-plan no matter how fast the replanning technique may be and hence alternative techniques must be sought.

Dynamical systems-based approaches to robot control offer such robustness to real-time perturbations. When controlled through a Dynamical System (DS), a robot motion unfolds in time with no need to re-plan. In this paper, we propose an obstacle avoidance algorithm that can be integrated into existing DS-based motion control approaches, while retaining the swiftness and robustness provided by these approaches. In the presented method, we assume that the robot motion is driven by a continuous and differentiable DS in the absence of obstacle(s). This DS is provided by the user, and henceforth we will call it the *original DS*. Given the original DS and an analytical formulation describing the surface of obstacles, our algorithm is able to instantly modify the robot's trajectory to avoid collisions with obstacles. Our approach has two main features: (1) As it only requires the differentiability of the original DS, it can be applied on a large set of DS including locally and globally asymptotically stable DS, autonomous and non-autonomous DS, limit cycles, unstable DS, etc., and (2) It does not modify the crit-

Electronic supplementary material The online version of this article (doi:10.1007/s10514-012-9287-y) contains supplementary material, which is available to authorized users.

S.M. Khansari-Zadeh (✉) · A. Billard
LASA Laboratory, School of Engineering, Ecole Polytechnique Federale de Lausanne (EPFL), 1015, Lausanne, Switzerland
e-mail: mohammad.khansari@epfl.ch

A. Billard
e-mail: aude.billard@epfl.ch

ical points of the original DS. Thus the attractors of the original DS are also the attractors of the modulated DS.

The rest of this paper is structured as follows. Section 2 describes main existing obstacle avoidance methods in the literature. Section 3 formalizes our obstacle avoidance algorithm for robot motions in the presence of a convex obstacle. Section 4 discusses the stability of the control law after applying the proposed obstacle avoidance algorithm. Section 5 describes how the avoidance trajectories can be customized through different parameters such as safety factor, reactivity, etc. Section 6 extends the presented approach to avoid multiple obstacles. Section 7 gives a conceptual sketch on how to use the proposed algorithm in robot experiments. Section 8 presents the experimental results, and Sect. 9 concludes the paper.

2 Related work

Obstacle avoidance is a classical problem in robotics and many approaches have been proposed to solve it. One may distinguish between local and global methods, depending on whether the obstacle influences the behavior only locally or everywhere. Local methods such as the Bug's algorithm (Lumelsky and Skewis 1990), the Vector Field Histogram (Borenstein and Koren 1991), and the Curvature-Velocity method (Simmons 1996) offer fast response in the face of perturbations. These are usually locally optimal and hence are not ensured to always find a feasible path.

Global methods, such as those dealt with by path planning algorithms (Lozano-Perez 1983; Kuffner and LaValle 2000; Kavraki et al. 1996) ensures to find a valid solution, if it exists. Despite recent efforts at reducing the computational costs of such global searches for a feasible path (Diankov and Kuffner 2007; Burns and Brock 2005; Toussaint 2009), these methods cannot offer the reactivity sought for swiftly avoiding obstacles that appear suddenly.

The reshaping method such as the Elastic Band approach (Quinlan and Khatib 1993; Brock and Khatib 2002) aims at realtime trajectory adaptation in dynamic environments. In this method, the initial shape of the elastic band is a free path generated by a classical planner. In the presence of obstacles, this band is deformed by applying repulsive forces. The work by Fraichard et al. (1991) also follows the same principle in which the original path is deformed locally to reflect changes in the environment topology. In these methods if the path being executed becomes infeasible due to obstacles coming into its way, the reshaping algorithm cannot be applied any more (Yoshida and Kanehiro 2011).

Hybrid systems that switch between local and global methods offer an interesting compromise. In Barbehenn et al. (1994), a task is decomposed into several segments that are amenable locally. If the local approach fails, the global method is invoked. Yoshida and Kanehiro (2011) propose a reactive motion planning approach which considers both the possibility of re-planning and deformation of the path during the execution of a task. In this approach, the planner first attempts to locally modify the trajectory in the presence of an obstacle. In situations where deformation is no longer possible (i.e. the path becomes infeasible), a new feasible trajectory is re-planned. The work by Vannoy and Xiao (2008) proposes an adaptive motion planner that considers the simultaneous path and trajectory planning of high-DOF robots. This method provides multiple diverse trajectories at all times to allow instant adaptation of robot motion to newly sensed changes in the environment. The elastic roadmap approach (Yang and Brock 2007) is similar to the conventional roadmap algorithm with the difference that it allows the modification of the vertices and edges during the execution of the task, hence the roadmap always represents task-consistent motions.

In Artificial Potential Fields (Khatib 1986) each obstacles is modeled with a repulsive force that prevents the robot from colliding with the obstacle. An appropriate repulsion force should be computed so that it repels sufficiently the trajectory away from the obstacle while avoiding to get stuck in local minima. The Attractor Dynamics Approach (Iossifidis and Schöner 2006) is another variant of the potential field method, which uses heading direction rather than the Cartesian position of the vehicle. The Dynamic Potential Field (Park et al. 2008) extends the potential field principle by taking into account not just the path but also the velocity along the path. Sprunk et al. (2011) propose a kinodynamic trajectory generation method, in which the dynamics of the robot is considered during path generation. This method uses quintic Bezier splines to specify position and orientation of the holonomic robot, and optimizes it according to a user-defined cost function.

Hoffmann et al. (2009) proposes a dynamical based approach to obstacle avoidance. This method, in essence, is very similar to the Attractor Dynamics approach in that it changes the original dynamics of motion by introducing a factor in the motion equation that stirs the motion away from the obstacle. This method is implemented to avoid point-mass objects in two and three dimensional spaces. For non-point objects, this approach requires determining a repulsion parameter that deforms the trajectory enough not to hit the obstacle.

Harmonic Potential functions (Kim and Khosla 1992; Feder and Slotine 1997) were first introduced to overcome the limitation of Potential Fields. This approach takes inspiration in the description of the dynamics of (incompressible and irrotational) fluids around impenetrable obstacles. In contrast to potential field-based methods, harmonic potential-based methods are powerful in that they do not have local minima. Harmonic potentials have been used for

control in numerous ways in the past few years. We mention here only the works that are closest to our method.

Kim and Khosla (1992) were among the first groups to use harmonic potential functions to control mobile robots and in particular to control a 3 DOF arm manipulator. Feder and Slotine (1997) extended Kim and Khosla's work to moving obstacles with constant translational and/or rotational velocities. To support multiple obstacles, they partitioned the space into regions affected by a single obstacle at most. To avoid the problem of partitioning, Waydo and Murray (2003) developed an alternative formulation using a continuous weighting factor. Similarly to Feder and Slotine (1997), this work only considered moving obstacles with constant velocity. A major advantage of harmonic potential functions over other potential functions is that they ensure that the target is the only attractor of the system. Unfortunately, in practice, requiring that the motions of both the robot and the obstacle follow harmonic functions may be too limiting.

In this paper we propose a local obstacle avoidance approach which can be used to locally modify the robot motions that are generated by a DS. The proposed method ensures that this local modification of trajectories does not change the main properties of the original DS. For instance, if the original DS is globally stable (i.e. all trajectories reach the target point) when there is no obstacle in the robot working space, it also remains stable in the presence of obstacles. The system described above could also be pictured as a hybrid controller in the sense that: the globally stable DS is the global planner generating trajectories that always reach the target, and the local planner is the proposed method that deforms the generated trajectory in the presence of obstacles. Both the path generation and deformation are done simultaneously at each time step. This approach is similar, in spirit, to the harmonic potential functions. The main differences lies in that our approach does not require the robot to follow harmonic functions, hence it can be applied to a larger set of robot motions.

3 Obstacle avoidance formulation

Consider a state variable $\xi \in \mathbb{R}^d$ that defines the state of a robotic system. Its temporal evolution may be governed by either an autonomous (time-invariant) or non-autonomous (time-varying) DS according to:

$$\dot{\xi} = f(\xi), \qquad f : \mathbb{R}^d \mapsto \mathbb{R}^d \qquad \text{autonomous DS} \quad (1)$$

$$\dot{\xi} = f(t, \xi), \qquad f : \mathbb{R}^+ \times \mathbb{R}^d \mapsto \mathbb{R}^d \qquad \text{non-auto. DS} \quad (2)$$

where $f(.)$ is a continuous function (we further use the notation $f(.)$ to refer to both autonomous and non-autonomous DS). Given an initial point $\{\xi\}_0$, the robot motion along time can be computed by integrating $f(.)$ recursively:

$$\{\xi\}_t = \{\xi\}_{t-1} + f(.)\delta t \quad (3)$$

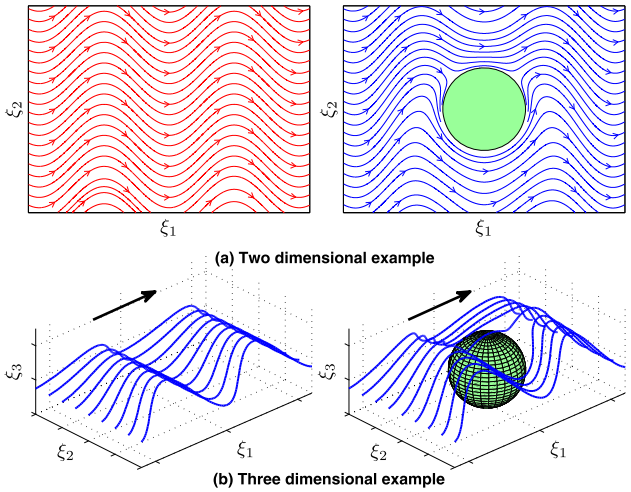


Fig. 1 Effect of the modulation induced by a spherical obstacle (located at the origin and with radius $r^o = 2$) on (**a**) a two dimensional flow generated by $\dot{\xi}_1 = 1.0$ and $\dot{\xi}_2 = \sin(\xi_1)$, and (**b**) a three dimensional flow generated by $\dot{\xi}_1 = 1.0$, $\dot{\xi}_2 = -\sin(\xi_2/4)\sin\xi_1$, and $\dot{\xi}_3 = \sin\xi_1$

where δt is the integration time step and t is a positive integer. Figures 1 and 3 illustrate a few examples of such functions.

Next we show how we can induce a modulation on our generic motion due to the presence of an obstacle. We first consider a hyper-sphere obstacle. We then extend this model to convex objects.

3.1 Hyper-sphere obstacles

Consider a d-dimensional hyper-sphere object centered at ξ^o with radius r^o. The object creates a modulation throughout the robot's state space, which is conveyed through the non-linear function $\phi^s(\xi; \xi^o, r^o) : \mathbb{R}^d \mapsto \mathbb{R}^d$ as follows:[1]

$$\phi^s(\xi; \xi^o, r^o) = \left(1 + \frac{(r^o)^2}{(\xi - \xi^o)^T(\xi - \xi^o)}\right)(\xi - \xi^o) \quad (4)$$

where $(.)^T$ denotes the transpose. To determine how ϕ modulates the velocity of the robot, we compute the Jacobian which yields:

$$M^s(\xi; \xi^o, r^o) = \nabla\phi^s(\xi; \xi^o, r^o) \quad (5)$$

[1] The development of Eq. (4) was partly inspired by the complex potential function that models the uniform flow around a circular cylinder (Milne-Thomson 1960). In both formulations the modulation of the flow due to the object's presence decreases quadratically with the distance to the center of the object (see the second term in Eq. (4)). The main difference between the two approaches lies in their functionality. Equation (4) is a d-dimensional vector and its Jacobian is a $d \times d$ matrix which can be used to modulate the original flow. In contrast, the complex potential function is a scalar value, and its derivative directly gives the modified flow in the presence of the obstacle.

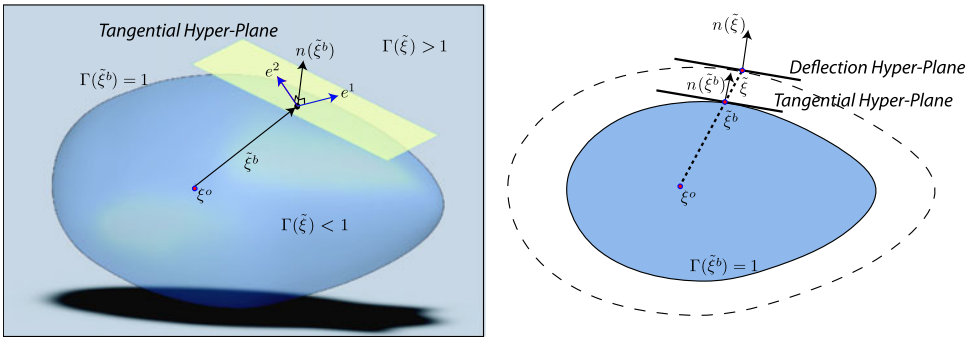


Fig. 2 Illustration of the tangential hyper-plane and its basis (*left*), and the deflection hyper-plane (*right*) for a 3-dimensional object

To simplify the notation, we express the modulation in a frame of reference centered on the object and define $\tilde{\xi} = \xi - \xi^o$:

$$M^s(\tilde{\xi}; r^o) = \boldsymbol{I} + \left(\frac{r^o}{\tilde{\xi}^T\tilde{\xi}}\right)^2 \left(\tilde{\xi}^T\tilde{\xi}\boldsymbol{I} - 2\tilde{\xi}\tilde{\xi}^T\right) \tag{6}$$

where $\boldsymbol{I}$ is the identity matrix. We call M^s the dynamic modulation matrix. The final model for real-time avoidance of spherical obstacles can be obtained by applying the dynamic modulation matrix to the original DS given by Eqs. (1)–(2):

$$\dot{\xi} = M^s(\tilde{\xi}; r^o) f(.) \tag{7}$$

$M^s(\tilde{\xi}; r^o)$ in Eq. (7) is a modulation factor that locally deforms the original dynamics f such that the robot does not hit the obstacle.

Theorem 1 *Consider a d-dimensional static hyper-sphere obstacle in $\mathbb{R}^d$ with center ξ^o and radius r^o. The obstacle boundary consists of the hyper-surface $\mathscr{X}^b \subset \mathbb{R}^d = \{\xi \in \mathbb{R}^d : \|\xi - \xi^o\| = r^o\}$. Any motion $\{\xi\}_t$, $t = 0..\infty$ that starts outside the obstacle, i.e. $\|\{\xi\}_0 - \xi^o\| > r^o$, and evolves according to Eq. (7) never penetrates into the obstacle, i.e. $\|\{\xi\}_t - \xi^o\| \geq r^o$.*

Proof See Appendix A. □

Figure 1 illustrates the effect of the modulation induced by such a spherical object on two and three-dimensional flows. As it is illustrated, in both cases the flow is deflected properly and it passes the obstacle.

3.2 Convex obstacles

Suppose a continuous function $\Gamma(\tilde{\xi})$ that projects $\mathbb{R}^d$ into $\mathbb{R}$. The function $\Gamma(\tilde{\xi})$ has continuous first order partial derivatives (i.e. C^1 smoothness) and increases monotonically with $\|\tilde{\xi}\|$. The level curves of Γ (i.e. $\Gamma(\tilde{\xi}) = c$, $\forall c \in \mathbb{R}$) enclose a convex region. By construction, the following relation holds at the surface of the obstacle:

$$\Gamma(\tilde{\xi}) = 1 \tag{8}$$

For example $\Gamma(\tilde{\xi}) : \sum_{i=1}^d (\tilde{\xi}_i/a_i)^2 = 1$ corresponds to a d-dimensional ellipsoid with axis lengths a_i. We can divide the space spanned by Γ into three regions $\mathscr{X}^o$, $\mathscr{X}^b$, and $\mathscr{X}^f$ to distinguish between points inside the obstacle, at its boundary, and outside the obstacle respectively:

$$\text{Interior points :} \quad \mathscr{X}^o = \{\xi \in \mathbb{R}^d : \Gamma(\tilde{\xi}) < 1\} \tag{9}$$

$$\text{Boundary points :} \quad \mathscr{X}^b = \{\xi \in \mathbb{R}^d : \Gamma(\tilde{\xi}) = 1\} \tag{10}$$

$$\text{Free region :} \quad \mathscr{X}^f = \{\xi \in \mathbb{R}^d : \Gamma(\tilde{\xi}) > 1\} \tag{11}$$

At each point $\xi^b \in \mathscr{X}^b$ on the outer surface of the obstacle, we can compute a tangential hyper-plane defined by its normal vector $n(\tilde{\xi}^b)$:

$$n(\tilde{\xi}^b) = \left[\frac{\partial\Gamma(\tilde{\xi}^b)}{\partial\tilde{\xi}^b_1} \cdots \frac{\partial\Gamma(\tilde{\xi}^b)}{\partial\tilde{\xi}^b_d}\right]^T \tag{12}$$

By extension, we can compute a *deflection hyperplane* at each point $\xi \in \mathscr{X}^f$ outside the obstacle with normal:

$$n(\tilde{\xi}) = \left[\frac{\partial\Gamma(\tilde{\xi})}{\partial\tilde{\xi}_1} \cdots \frac{\partial\Gamma(\tilde{\xi})}{\partial\tilde{\xi}_d}\right]^T \tag{13}$$

Each point on the deflection hyper-plane can be expressed as a linear combination of a set of $(d-1)$ linearly independent vectors. These vectors form a basis of the deflection hyper-plane. One particular set of such vectors $e^1, \ldots, e^{d-1}$ is[2]

$$e^i_j(\tilde{\xi}) = \begin{cases} -\frac{\partial\Gamma(\tilde{\xi})}{\partial\tilde{\xi}_i} & j = 1 \\ \frac{\partial\Gamma(\tilde{\xi})}{\partial\tilde{\xi}_1} & j = i \neq 1 \\ 0 & j \neq 1,\ j \neq i \end{cases} \qquad i \in 1..d-1\ ,\ j \in 1..d \tag{14}$$

where e^i_j corresponds to the j-th component of the i-th basis vector. Figure 2 illustrates the tangential and the deflection hyper-planes for a three-dimensional object.

[2] In case $\partial\Gamma(\tilde{\xi})/\partial\tilde{\xi}_1$ vanishes, the vectors are no longer linearly independent and one should choose another index for the derivative which is non-zero.

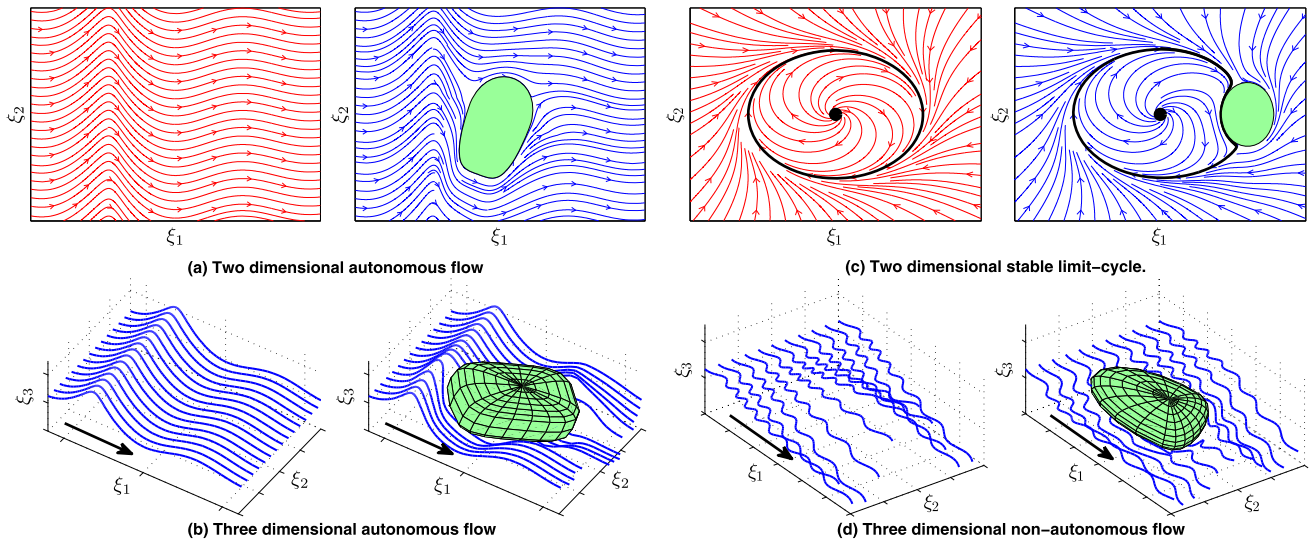


Fig. 3 Modifying the original motion of a flow with a modulation matrix for: (**a**) A two dimensional flow with $\dot{\xi}_1 = \log((\xi_1+3)^2+2)$ and $\dot{\xi}_2 = \sin(\xi_1)$. (**b**) A three dimensional autonomous flow with $\dot{\xi}_1 = \log((\xi_1+3)^2+2)$, $\dot{\xi}_2 = 0$, and $\dot{\xi}_3 = \sin(\xi_1)$. (**c**) A stable limit cycle motion with $\dot{\xi}_1 = \xi_2 - \xi_1(\xi_1^2+\xi_2^2-1)$ and $\dot{\xi}_2 = -\xi_1 - \xi_2(\xi_1^2+\xi_2^2-1)$. (**d**) A three dimensional non-autonomous flow with $\dot{\xi}_1 = \log((\xi_1+3)^2/(t+1)+2)$, $\dot{\xi}_2 = \sin(5t)-0.1$, and $\dot{\xi}_3 = 0.05t\cos(\xi_2)$. In all four cases the obstacle is centered at $\xi^o = \mathbf{0}$. In (**c**), the *thick black line* represents the stable limit cycle

As in the case of the spherical object, we can determine a modulation matrix $M(\tilde{\xi})$ given by:[3]

$$M(\tilde{\xi}) = E(\tilde{\xi})\, D(\tilde{\xi})\, E(\tilde{\xi})^{(-1)} \tag{15}$$

with the matrices of basis vectors $E(\tilde{\xi})$ and associated eigenvalues $D(\tilde{\xi})$:

$$E(\tilde{\xi}) = \begin{bmatrix} n(\tilde{\xi}) & e^1(\tilde{\xi}) & \cdots & e^{d-1}(\tilde{\xi}) \end{bmatrix} \tag{16}$$

$$D(\tilde{\xi}) = \begin{bmatrix} \lambda^1(\tilde{\xi}) & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & \lambda^d(\tilde{\xi}) \end{bmatrix} \tag{17}$$

where

$$\begin{cases} \lambda^1(\tilde{\xi}) = 1 - \frac{1}{|\Gamma(\tilde{\xi})|} \\ \lambda^i(\tilde{\xi}) = 1 + \frac{1}{|\Gamma(\tilde{\xi})|} \quad 2 \le i \le d \end{cases} \tag{18}$$

The dynamic modulation matrix $M(\tilde{\xi})$ propagates the influence of the obstacle on the motion flow. The result of Eq. (15) is invariant to the choice of the basis $e^1..e^{d-1}$. Furthermore, the matrix of basis vector is invertible in $\mathbb{R}^d \setminus \xi^o$. At the obstacle reference point ξ^o, the deflection hyperplane is undefined; however, this does not cause any problem since ξ^o is a point inside the obstacle (recall $\Gamma(\mathbf{0}) < 1$). Moreover, since $\Gamma(\tilde{\xi})$ monotonically increases with $\|\tilde{\xi}\|$, the matrix of eigenvalues and by extension the dynamic modulation matrix converge to the identity matrix as the distance to the obstacle increases. Hence, the effect of the dynamic modulation matrix is maximum at the boundaries of the obstacle, and vanishes for points far from it.

Similarly to the hyper-sphere obstacle avoidance given by Eq. (7), we can apply the modulation given by Eq. (15) on our original motion flow f which yields:

$$\dot{\xi} = M(\tilde{\xi})\, f(.) \tag{19}$$

Theorem 2 *Consider a convex manifold $\Gamma(\tilde{\xi}) = 1$ that encloses a static d-dimensional obstacle with respect to a reference point ξ^o inside the obstacle. A motion $\{\xi\}_t$, that starts outside the obstacle, i.e. $\Gamma(\{\xi\}_0) \ge 1$, and evolves according to Eq. (19) does not penetrate the obstacle, i.e. $\Gamma(\{\xi\}_t) \ge 1$, $t = 0..\infty$.*

Proof See Appendix B. □

Figure 3 illustrates with four examples the effect of the modulation induced on the field of motion in the presence of different obstacles.

4 Robot discrete movements

So far we have shown how the dynamic modulation matrix $M(\tilde{\xi})$ can be used to deform a robot motion such that it does not collide with an obstacle. However in many robot experiments, e.g. reaching a target, not only should the robot avoid

[3] Derivation of Eqs. (15)–(16) are inspired from the proof of Theorem 1. For a spherical obstacle, these equations yield to the same result given by Eq. (6).

the obstacle, but it should also reach a target, which we further denote ξ^*. In other words, we would like the modified motion to preserve the convergence property of the original dynamics while still ensuring that the motion does not penetrate the object. In this section we discuss the stability of DS when they are modulated with the proposed obstacle avoidance method. Throughout the section, we will assume that the target point ξ^* is outside the obstacle boundary, i.e. $\xi^* \in \mathscr{X}^f$.

Suppose a d-dimensional globally asymptotically stable autonomous or non-autonomous DS defined by Eqs. (1) or (2). The global stability of f requires that the velocity vanishes solely at the target point ξ^*, i.e. $f(\xi^*) = 0$ for autonomous DS and $\lim_{t\to\infty} f(t, \xi^*) = 0$ for non-autonomous DS. When f is modulated with the dynamic modulation matrix $M(\tilde{\xi})$, ξ^* remains an equilibrium point because the velocity still vanishes at the target, i.e. $M(\xi^* - \xi^o) f(\xi^*) = 0$ for autonomous DS, and $\lim_{t\to\infty} M(\xi^* - \xi^o) f(t, \xi^*) = M(\xi^* - \xi^o) \lim_{t\to\infty} f(t, \xi^*) = 0$ for non-autonomous DS.

However, in the presence of an obstacle, the target may not remain the unique equilibrium point of the system. Other possible equilibrium points may be created due to the modulation term $M(\tilde{\xi})$. These points can be computed by looking at the null space of $M(\tilde{\xi})$. For all $\xi \in \mathscr{X}^f$, the matrix $M(\tilde{\xi})$ is full rank and hence ξ^* will be *the only equilibrium point* in $\mathscr{X}^f$. Only on the boundaries of the obstacle, i.e. $\xi^b \in \mathscr{X}^b$, $M(\tilde{\xi}^b)$ loses one rank yielding a number of spurious equilibrium points. In fact, these spurious equilibrium points $\xi^s \in \mathscr{X}^b$ are generated when there is collinearity between the velocity and the normal vector at the boundary points:[4]

$$n\left(\tilde{\xi}^s\right)^T \frac{f(.)}{\|f(.)\|} = \pm 1 \quad \text{and} \quad \Gamma\left(\tilde{\xi}^s\right) = 1 \tag{20}$$

where $n(\tilde{\xi}^s)$ is the unit normal vector of the tangential hyperplane at $\tilde{\xi}^s$. The set $\mathscr{X}^s$ includes all solutions to Eq. (20). Depending on the function f, these equilibrium points could be either saddle points and/or local minima.

Computing this set of equilibrium points may not always be feasible. We can however simplify our task by observing that, since all the equilibrium points appear solely on the obstacle boundary, one may avoid remaining stuck by using some external mechanisms. Algorithm 1 describes such a mechanism: when one detects that the motion has stopped at the outer surface (boundary) of an obstacle (i.e. at an equilibrium point), she applies a small perturbation along any of the basis vectors $e^1..e^{d-1}$. All of these vectors determine directions that ensure that the flow will move away from the obstacle. If the equilibrium point is a saddle point, the algorithm exits in one iteration. But if it is a local minimum, the obstacle is contoured along the direction of the basis vector e^i until it leaves the basin of attraction of the local minimum. The positive scalar α controls the amplitude of the movement along the basis vector e^i. The value of α should be chosen by compromising between the accuracy, safety, and speed of the movement. For large integration time step δt, one should use a small α to decrease the drifting error (due to integration) from the desired trajectory when contouring the obstacle. Furthermore, since contouring takes place at the outer surface of the obstacle, for safety reasons one should generally avoid selecting a high value for α. A very small value for α is also not recommended since it significantly slows down the contouring speed. Figure 4 illustrates two examples where the Algorithm 1 is used to handle a saddle point and a local minimum.

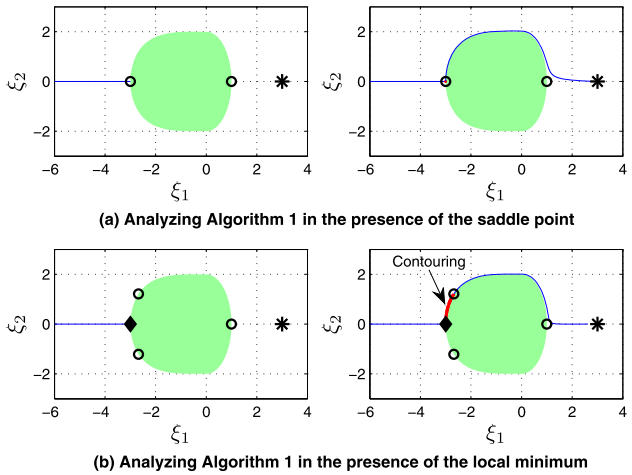


Fig. 4 Illustration of using Algorithm 1 to avoid possible equilibrium point(s) on the obstacle boundary. The target point is shown with a black star. The saddle point(s) and local minimum are represented with hollow circle and diamond, respectively. The obstacle boundary is modeled with $(\tilde{\xi}_1/1)^2 + (\tilde{\xi}_2/2)^2 = 1$ when $\tilde{\xi}_1 > 0$ and $(\tilde{\xi}_1/3)^4 + (\tilde{\xi}_2/2)^2 = 1$ elsewhere. (**a**) When the DS is defined by $\dot{\xi}_1 = -\xi_1 + 3$ and $\dot{\xi}_2 = -\xi_2$, the modulated dynamics has two saddle points at $(-3, 0)$ and $(0, 1)$. Without using Algorithm 1, the motion stops at $(-3, 0)$ (see (**a**)-*left*). However, by using Algorithm 1 for one iteration, the motion continues until it reaches the target (see (**a**)-*right*). (**b**) By modifying the DS along its second dimension to $\dot{\xi}_2 = -3\xi_2$, the modulated dynamics will have one local minimum at $(-3, 0)$ and three saddle point at $(0, 1)$, $(-2.6757, 1.2120)$, and $(-2.6757, -1.2120)$. Without using Algorithm 1, the motion stops at the local minimum $(-3, 0)$ (see (**b**)-*left*). In this situation, Algorithm 1 is used iteratively until the trajectory leaves the basin of attraction of the local minimum (i.e. the range between the local minimum and the saddle point). Then, the motion continues its way to the target (see (**b**)-*right*). The part of trajectory that generated by Algorithm 1 is plotted with a *thick red line*

5 Characterizing the path during obstacle avoidance

When doing obstacle avoidance, sometimes it is more practical to customize the path to avoid an obstacle based on the

[4] From Theorem 2 we know that the normal velocity at the boundary points vanishes. Hence, if $f(\xi)$ is aligned with the normal vector of the tangential hyperplane at a boundary point, we have $M(\tilde{\xi}) f(\xi) = 0$.

Algorithm 1 Procedure to handle equilibrium points at the obstacle boundary

Require: ξ^t, $\dot{\xi}^t$, and the integration time step δt
1: **if** $\Gamma(\tilde{\xi}^t) = 1$ and $\dot{\xi}^t = 0$ **then**
2: Choose one of the basis vectors e^i of tangential hyper-plane.
3: Define a small positive scalar $\alpha > 0$
4: **while** true **do**
5: $\xi^{t+1} \leftarrow \xi^t + \alpha e^i \delta t$
6: Compute $\dot{\xi}^{t+1}$ from Eq. (19)
7: **if** $(e^i)^T \dot{\xi}^{t+1} > 0$ or $n(\tilde{\xi})^T \dot{\xi}^{t+1} > 0$ **then**
8: exit
9: **end if**
10: $t \leftarrow t + 1$
11: **end while**
12: **end if**

object's property. For example, fragile or sharp objects may require a large safety margin while soft and round object may not. Furthermore, it is essential to react and deflect the robot trajectory earlier when it goes toward a fire flame than when it is just heading towards a soft pillow. In this section, we extend the proposed obstacle avoidance approach to incorporate user's preference during obstacle avoidance.

5.1 Safety margin

The desired safety margin around an object can be obtained by scaling the state variable (in the obstacle frame of reference) in the dynamic modulation matrix $M(\tilde{\xi})$ given by Eq. (18) as follows:

$$M(\tilde{\xi}_\eta) = E(\tilde{\xi}_\eta)\, D(\tilde{\xi}_\eta)\, E(\tilde{\xi}_\eta)^{(-1)} \tag{21}$$

where $\tilde{\xi}_\eta = \tilde{\xi}./\eta$ corresponds to the element-wise division of $\tilde{\xi}$ by $\eta \in \mathbb{R}^d$, and $\eta_i \geq 1$, $\forall i \in 1..d$ is the desired safety factor, which inflates the object along each direction $\tilde{\xi}_1$ with the magnitude η_i (in the obstacle frame of reference). By choosing different value for each η_i, one can control the required safety margin along the corresponding direction of the object. Figure 5 illustrates the effect of different safety margins for a 2D object in a uniform flow.[5]

5.2 Reactivity

The magnitude of the modulation created by the obstacle can be tuned by modifying the eigenvalues of the dynamic modulation matrix as follows:

$$\begin{cases} \lambda^1(\tilde{\xi}) = 1 - \frac{1}{|\Gamma(\tilde{\xi})|^{\frac{1}{\rho}}} \\ \lambda^i(\tilde{\xi}) = 1 + \frac{1}{|\Gamma(\tilde{\xi})|^{\frac{1}{\rho}}} \quad 2 \leq i \leq d \end{cases} \tag{22}$$

where $\rho > 0$ is the reactivity parameter. The larger the reactivity, the larger the amplitude of the deflection, and consequently the earlier the robot responds to the presence of an obstacle. A large ρ also extends the deflection farther out. Figure 6 illustrates the effect of using different reactivity parameters for a 2D object in a uniform flow.

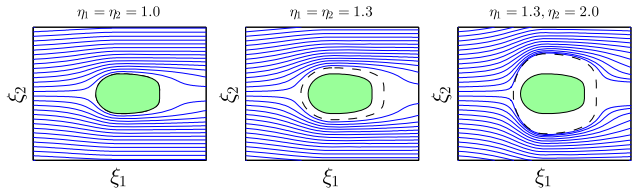


Fig. 5 Controlling the safety margin around the obstacle via the safety factor. The obstacle is inflated in the direction ξ_1 and ξ_2 with the value η_1 and η_2, respectively. The area between the *dashed line* and the obstacle boundary is the safety margin. The direction of the motion is from *left* to *right*

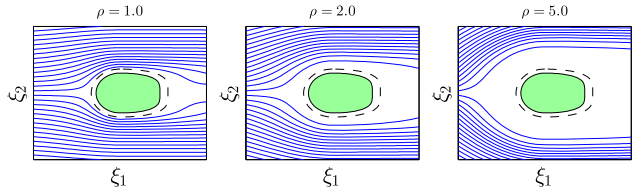


Fig. 6 Controlling the reactivity of the motion to the presence of the obstacle (for $\eta_1 = \eta_2 = 1.2$). By increasing ρ, the reactivity increases, hence the flow deflects earlier in time and with a higher magnitude. Note that on the right graph, the white gap between the *dashed line* and the trajectories is part of the free region

5.3 Tail-effect

In the proposed obstacle avoidance formulation, the modulation due to the obstacle continues affecting the motion even when the robot is moving away from the obstacle (see Fig. 7-left). We call this effect of the obstacle on trajectories *tail-effect*. In case of uncertainty in sensing, such a behavior may be beneficial as it would mitigate imprecise detection of the real volume of the obstacle. When it is not desirable, one can remedy the tail-effect by defining the first eigenvalue of the dynamic modulation matrix as follows:

$$\lambda^1(\tilde{\xi}) = \begin{cases} 1 - \frac{1}{|\Gamma(\tilde{\xi})|^{\frac{1}{\rho}}} & n(\tilde{\xi})^T \dot{\xi} < 0 \\ 1 & n(\tilde{\xi})^T \dot{\xi} \geq 0 \end{cases} \tag{23}$$

In the above equation, we use the sign of $n(\tilde{\xi})^T \dot{\xi}$ to check whether a trajectory is going towards (negative sign) or away

[5] One can also define different safety factors along the positive and negative directions of each object's axis by considering an *if-else* condition on the sign of each $\tilde{\xi}_i$.

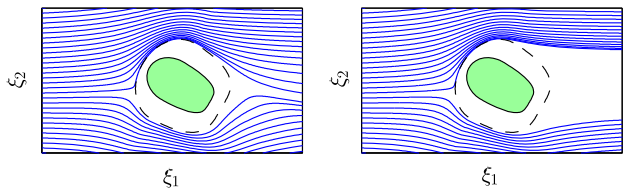


Fig. 7 Controlling the tail-effect after passing the obstacle. *Left:* The tendency of the trajectories to follow the obstacle shape after passing it. *Right:* Remedying the tail-effect by defining the first eigenvalue according to Eq. (23)

(positive sign) from the obstacle. Figure 7-right illustrates the result after using Eq. (23). In this figure one can see that the tail-effect is significantly reduced. However, the slight modulation of the trajectories after passing the obstacle is still required in order to ensure the continuity in the velocity.

6 Extension to multiple obstacles

So far we have shown how the dynamic modulation matrix can be used to avoid a single obstacle. However, in the presence of multiple obstacles, the current dynamic modulation matrix is ineffective and should be modified to include the effect of all the obstacles. Beware that this extension *cannot* be simply obtained by multiplying together the dynamic modulation matrix of all the obstacles. In this case, the impenetrability condition is only guaranteed for one of the obstacles. Note that for the sake of clarity of equations, in this section we did not consider the extensions that we have provided in Sect. 5 on the safety margin, reactivity, and tail-effect (here we use the default value $\eta = \rho = 1$, and do not remedy the tail-effect). In Sect. 7, we unify all these extensions into a single final model (see Table 1).

Let us consider K obstacles with associated reference points $\xi^{o,k}$ and boundary functions $\Gamma^k(\xi;\xi^{o,k})$, $k = 1..K$ (the parameters of the k-th obstacle is denoted by $(.)^k$). We modify Eq. (18), and compute the eigenvalues of the k-th obstacle based on both its current state, and the state of other obstacles as follows:

$$\begin{cases} \lambda_1^k(\tilde{\xi}^k) = 1 - \dfrac{\omega^k(\tilde{\xi}^k)}{|\Gamma(\tilde{\xi}^k)|} \\ \lambda_i^k(\tilde{\xi}^k) = 1 + \dfrac{\omega^k(\tilde{\xi}^k)}{|\Gamma(\tilde{\xi}^k)|} \quad 2 \le i \le d \end{cases} \tag{24}$$

where $\tilde{\xi}^k = \xi - \xi^{o,k}$, $\Gamma^k(\xi^k)$ is the simplified notation of $\Gamma^k(\xi;\xi^{o,k})$, and $\omega^k(\tilde{\xi}^k)$ are weighting coefficients that are computed according to:[6]

$$\omega^k(\tilde{\xi}^k) = \prod_{i=1, i\neq k}^{K} \frac{(\Gamma^i(\tilde{\xi}^i) - 1)}{(\Gamma^k(\tilde{\xi}^k) - 1) + (\Gamma^i(\tilde{\xi}^i) - 1)} \tag{25}$$

First observe that $\omega^k(\tilde{\xi}^k)$ are continuous positive scalars between zero and one, i.e. $0 \le \omega^k(\tilde{\xi}^k) \le 1$. Second, at the boundary of the k-th obstacle (i.e. $\Gamma^k(\tilde{\xi}^k) = 1$), we have $\omega^k(\tilde{\xi}^k) = 1$ and $\omega^i(\tilde{\xi}^i) = 0, \forall i \in 1..K$ and $i \neq k$. As we will discuss later on, these two properties are crucial to ensure impenetrability of the obstacles. Note that, when only one obstacle exists ($K = 1$), we simply set $\omega^1(\tilde{\xi}^1) = 1$ and Eq. (24) simplified into Eq. (18).

By substituting Eq. (25) into the matrix of eigenvalues given by Eq. (17), the dynamic modulation matrix for each obstacle becomes:

$$M^k(\tilde{\xi}^k) = E^k(\tilde{\xi}^k)\, D^k(\tilde{\xi}^k)\, \left(E^k(\tilde{\xi}^k)\right)^{-1} \tag{26}$$

The combined modulation matrix that considers the net effect of all the obstacles is then given by:

$$\bar{M}(\xi) = \prod_{k=1}^{K} M^k(\tilde{\xi}^k) \tag{27}$$

Equation (27) ensures the impenetrability of all the K obstacles. To verify this, suppose a point ξ^b on the boundary of the k-th obstacle. At this point, following the properties of ω mentioned above and considering Eqs. (24), (17), (26), and (27), we have:

$$\begin{aligned} \omega^i(\tilde{\xi}^{b,i}) = 0 \quad &\Rightarrow \quad \lambda_j^i(\tilde{\xi}^{b,i}) = 1 \quad \forall j \in 1..d,\ \forall i \in 1..K, \\ &\qquad\qquad\qquad\qquad i \neq k \\ &\Rightarrow \quad D^i(\tilde{\xi}^{b,i}) = \boldsymbol{I} \\ &\Rightarrow \quad M^i(\tilde{\xi}^{b,i}) = E^i(\tilde{\xi}^{b,i})\, \boldsymbol{I}\, \left(E^i(\tilde{\xi}^{b,i})\right)^{-1} \\ &\qquad\qquad\qquad = \boldsymbol{I} \\ &\Rightarrow \quad \bar{M}(\xi^b) = M^k(\tilde{\xi}^{b,k}) \end{aligned}$$

Furthermore, because $\omega^k(\tilde{\xi}^{b,k}) = 1$, $M^k(\tilde{\xi}^{b,k})$ and by extension $\bar{M}(\xi^b)$ is exactly similar to Eq. (15). Hence following Theorem 2, the obstacle is impenetrable. By moving from one obstacle to another, the weighting coefficients smoothly changes between zero and one, and by this, impenetrability is always ensured for all the obstacles.

Following the discussion given in Sect. 4, the target point ξ^* is the only equilibrium point in the free region because

[6] Equation (25) is in spirit very similar to the weighting coefficients proposed in Waydo and Murray (2003) with the difference that we use $\Gamma^k(\xi)$ to compute weights (rather than the distance between the obstacles).

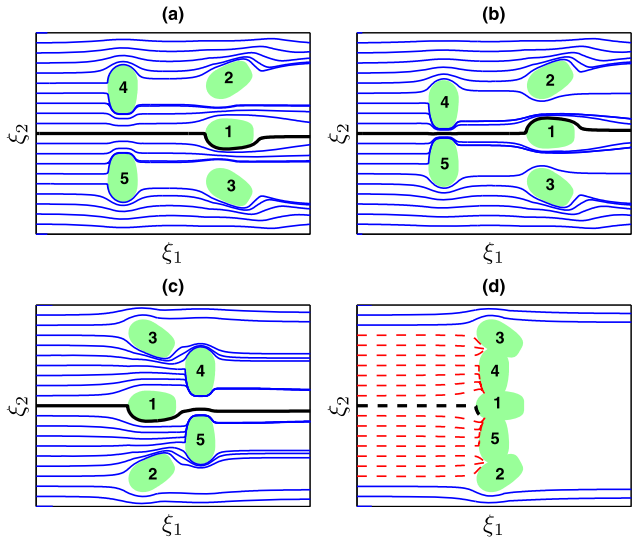


Fig. 8 Extension of the proposed approach to multiple obstacles. The combined dynamic modulation matrix ensures the impenetrability of all obstacle even if they are very close or connected to each other. However, for the case where the objects are connected (see (**d**)), some local minima may appear that cannot be avoided with Algorithm 1. Trajectories that stop at the local minima are plotted with dashed lines. A trivial solution to handle this problem is to model all the connected obstacles as a single convex obstacle

Fig. 9 Illustration of two complex objects that are modeled with two smooth hyper-surfaces. The analytical model for the drawer is $\Gamma(\tilde{\xi})$: $(\tilde{\xi}_1/0.4)^4 + (\tilde{\xi}_2/0.4)^8 + (\tilde{\xi}_3/0.6)^4 = 1$, and the mug is modeled with $(\tilde{\xi}_1/0.05)^4 + (\tilde{\xi}_2/0.05)^8 + (\tilde{\xi}_3/0.05)^4 = 1$ when $\tilde{\xi}_2 > 0$ and $(\tilde{\xi}_1/0.05)^4 + (\tilde{\xi}_2/0.08)^2 + (\tilde{\xi}_3/0.05)^4 = 1$ elsewhere

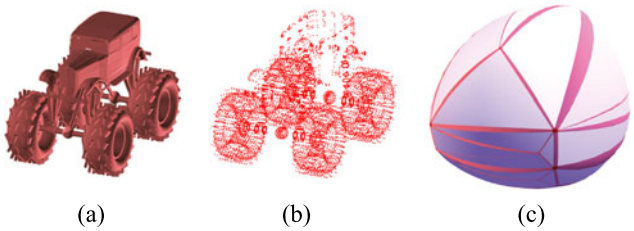


Fig. 10 Illustration of generating a BV from the point cloud of a toy car. (**a**) The 3D model of the car. (**b**) The point cloud of the car taken from the Princeton Shape Benchmark (Shilane et al. 2004). (**c**) The $C1$ smoothness BV generated using the method described by Benallegue et al. (2009)

all the modulation matrices M^k has full rank. However, as discussed before, on the boundaries of each obstacle a set of saddle points or local minima may be generated. Provided the obstacles are not connected, i.e. they do not have a contact point, these equilibrium points can be handled by following Algorithm 1.

Figure 8 illustrates the implementation of Eq. (27) in the presence of five obstacles positioned in different ways. To simplify the reference to these objects, they are numbered from one to five. In this figure, the thick black line is the streamline that starts on the symmetric line of the obstacles arrangement. As can be seen, the combined modulation matrix is able to prevent hitting the obstacles even if there is a narrow passage between them (see for example Figs. 8(a), (b) or (c)).

Figure 8(d) shows the result for the case where all obstacles are connected. First observe that the resulting shape is no longer convex, but the impenetrability of the obstacles is still preserved. However in the presence of the resulting concave shape, Algorithm 1 cannot be used to avoid local minima. A trivial solution to handle this problem is to model all the connected obstacles as a single convex obstacle. Note that at the boundaries' intersection points, the weighting coefficients ω^k are undefined (because the distance to more than one obstacle is zero, and thus a division by zero occurs). At these points, we have simply stopped the simulation.

7 Obstacle avoidance module

The proposed obstacle avoidance algorithm requires a user to provide an analytical formulation of the outer surface of the obstacle. When provided with the 3D model of the object, one may compute a smooth convex envelope (also known as convex bounding volume) that fits tightly around the object. This Bounding Volume (BV) can be used (instead of the object's shape) to perform obstacle avoidance. Figure 9 illustrates such 3D convex envelopes generated from the 3D models of a mug and a drawer.

When solely the point cloud description of the object is available, one may use one of the estimation techniques to approximate the BV. For example, in Benallegue et al. (2009), the BV is approximated using a set of spheres and tori. To use this method, one first needs to find the relevant patch (either sphere or torus) of the BV that corresponds to the current position of the robot. Then, based on the analytical formulation of that patch, one can compute the dynamic modulation matrix as described before. Recall that our obstacle avoidance module only requires the convexity and $C1$ smoothness of the BV, which are fulfilled in this work. Figure 10 shows an example of the convex BV generated from the point cloud of a toy car using the method above.

When doing obstacle avoidance in a dynamic environment, it is hardly possible to generate the BVs from the output of the vision system in realtime. Thus, it is necessary to

Table 1 The complete formulation of dynamic modulation matrix

Nomenclature		Formulation
d	Dimension of state variable	**(a)** For each obstacle compute the followings:
K	Number of Obstacles	**(a.1)** $\tilde{\xi}_\eta^k = (\xi - \xi^{o,k})./\eta$
$\xi \in \mathbb{R}^d$	Current robot position	**(a.2)** $E^k(\tilde{\xi}_\eta^k) = \left[n^k(\tilde{\xi}_\eta^k) \quad e^{1,k}(\tilde{\xi}_\eta^k) \quad \cdots \quad e^{d-1,k}(\tilde{\xi}_\eta^k)\right]$
$\dot{\xi} \in \mathbb{R}^d$	Current robot velocity	**(a.3)** $\omega^k(\tilde{\xi}_\eta^k) = \prod_{i=1, i\neq k}^{K} \frac{(\Gamma^i(\tilde{\xi}_\eta^k) - 1)}{(\Gamma^k(\tilde{\xi}_\eta^k) - 1) + (\Gamma^i(\tilde{\xi}_\eta^k) - 1)}$
$\xi^{o,k} \in \mathbb{R}^d$	Center of k-th obstacle	**(a.4)** $\begin{cases} \lambda_1^k(\tilde{\xi}_\eta^k) = \begin{cases} 1 - \frac{\omega^k(\tilde{\xi}_\eta^k)}{\lvert\Gamma(\tilde{\xi}_\eta^k)\rvert^{\frac{1}{\rho}}} & n(\tilde{\xi})^T\dot{\xi} < 0 \text{ or } \kappa = 1 \\ 1 & n(\tilde{\xi})^T\dot{\xi} \geq 0 \text{ and } \kappa = 0 \end{cases} \\ \lambda_i^k(\tilde{\xi}_\eta^k) = 1 + \frac{\omega^k(\tilde{\xi}_\eta^k)}{\lvert\Gamma(\tilde{\xi}_\eta^k)\rvert^{\frac{1}{\rho}}} \qquad 2 \leq i \leq d \end{cases}$
$\tilde{\xi}^k \in \mathbb{R}^d$	Robot relative position to k-th obstacle	**(a.5)** $D(\tilde{\xi}_\eta^k) = \begin{bmatrix} \lambda_1^k(\tilde{\xi}_\eta^k) & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & \lambda_d^k(\tilde{\xi}_\eta^k) \end{bmatrix}$
$\tilde{\xi}_\eta^k \in \mathbb{R}^d$	Scaled robot relative position to k-th obstacle	**(a.6)** $M^k(\tilde{\xi}_\eta^k) = E^k(\tilde{\xi}_\eta^k)\, D^k(\tilde{\xi}_\eta^k)\, (E^k(\tilde{\xi}_\eta^k))^{-1}$
$\Gamma^k : \mathbb{R}^d \mapsto \mathbb{R}$	Analytical description of k-th obstacle	**(b)** Combined Dynamic Modulation Matrix: $\bar{M}(\xi) = \prod_{k=1}^{K} M^k(\tilde{\xi}_\eta^k)$
$E^k \in \mathbb{R}^{d\times d}$	Matrix of Basis vectors of k-th obstacle	
$D^k \in \mathbb{R}^{d\times d}$	Matrix of eigenvalues of k-th obstacle	
$M^k \in \mathbb{R}^{d\times d}$	Dynamic Modulation Matrix of k-th obstacle	
$n^k \in \mathbb{R}^d$	Normal vector of deflection hyperplane for k-th obstacle	
$e^{i,k} \in \mathbb{R}^d$	i-th basis vector of k-th obstacle	
$\lambda_i^k \in [0\ 2]$	i-th eigenvalue of k-th obstacle	
$\omega^k \in [0\ 1]$	Weighting coefficient of k-th obstacle	
$\eta \in [0\ \infty)$	Safety factor	
$\rho \in \mathbb{R}^+$	Reactivity	
$\kappa \in \{0, 1\}$	Tail-Effect	

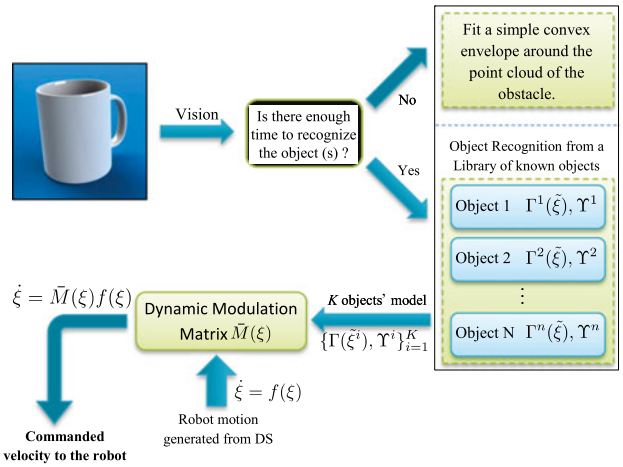


Fig. 11 A conceptual sketch describing the implementation of the obstacle avoidance module for robot motions. The set $\Upsilon^i = \{\eta^i, \rho^i, \kappa^i\}$ contains the user preference for each obstacle

generate a library that stores the analytical formulations of different objects. In our implementation, we rely on a library of objects with known analytical convex envelopes. We use this analytical descriptor of the envelop both to detect the object and for our obstacle avoidance module.

Figure 11 illustrates a conceptual sketch describing how the presented obstacle avoidance method can be used in robot experiments. In this approach, first the raw output of the vision system is sent to an object recognition module to identify the object(s). When the objects are recognized, their corresponding properties such as the analytical formulation of the boundary, safety factor, etc. are sent to the obstacle avoidance module. The obstacle avoidance module modifies the original dynamics of the motion by multiplying it with the combined dynamic modulation matrix $\bar{M}(\xi)$ so as to avoid the obstacle safely. The complete formulation of dynamic modulation matrix is summarized in Table 1.

In the presence of fast unknown moving obstacles, the object recognition phase may not provide the agility required to avoid the obstacle (especially when there is a large library of the objects). In these situations, it might be more adequate to replace the object recognition phase with an automatic BV generator algorithm (see Fig. 11). Generating a simple BV (e.g. an ellipsoid) around the point cloud of an obstacle can be done quite quickly. If the object moves very rapidly, it is recommended to set a large value for the safety margin η and for the reactivity parameter ρ (see Sect. 5) to increase the robustness to uncertainties.

Furthermore, when there are many obstacles in the working space of the robot, it may not be necessary (and also computationally feasible) to track all the obstacles all the time. Since the modulation decreases as the distance to the obstacle increases, one could ignore all obstacles for which

Table 2 The theoretical DS used for the Simulation Experiments

(a) $\begin{cases}\dot{x} = -x \\ \dot{y} = -x\cos x - y\end{cases}$	**(d)** $\begin{cases}\dot{x} = y - x(x^2 + y\sin x - 1) \\ \dot{y} = -x - y(x^2 + y\sin x - 1)\end{cases}$
(b) $\begin{cases}\dot{x} = \cos x \\ \dot{y} = \sin y\end{cases}$	**(e)** $\begin{cases}\dot{x} = \lvert x\rvert/2 + 1 \\ \dot{y} = 0 \\ \dot{z} = \lvert y\rvert\cos t\end{cases}$
(c) $\begin{cases}\dot{x} = y \\ \dot{y} = -x + 0.9y(1 - x^2)\end{cases}$	

the associated modulation matrices are close to identity[7] (since we have $\lim_{\tilde{\xi}^k \to \infty} M^k(\tilde{\xi}^k) = \boldsymbol{I}$).

By taking into account the obstacles that are locally relevant, the processing time for the vision systems could decrease significantly. However, this will be at the cost of imposing a small discontinuity in the robot velocity when an obstacle is added or removed from the set of relevant obstacles. By setting a small threshold, this discontinuity practically becomes very negligible.

8 Experiments

We evaluate the performance of the proposed approach in three ways: (1) On a set of theoretical autonomous and non-autonomous DS, (2) On a set of 2D motions described by dynamical systems that were inferred from human demonstrations, using two different learning approaches: Stable Estimator of Dynamical Systems (SEDS) (Khansari-Zadeh and Billard 2011) and Dynamic Movement Primitives (DMP) (Hoffmann et al. 2009) (see Sect. 8.2 for further information about these approaches), and (3) In five robot experiments performed on the 7-DOF Barrett WAM arm. Unless otherwise specified, throughout this section we consider $\rho = \kappa = 1$, and the state of the system is defined as either planar or 3D motions, i.e. $\xi = [x \;\; y]^T$ or $\xi = [x \;\; y \;\; z]^T$ respectively.

8.1 Simulation experiments on theoretical DS

We first evaluate the method in simulation using our basic motion flow $f(.)$ for five different dynamical systems. These DS are defined in Table 2 and their phase plots are illustrated in Fig. 12.

The first DS is globally asymptotically stable at the origin. Due to the cosine term, this DS displays a high nonlinear behavior. The second DS is interesting in that it has infinite number of attractors, saddle points, and unstable points.

[7]For example, we consider the k-th obstacle is locally relevant in the current position of the robot if: $|\lambda_i^k(\tilde{\xi}^k) - 1| > \varsigma, \forall i = 1..d$, where ς is a small positive threshold.

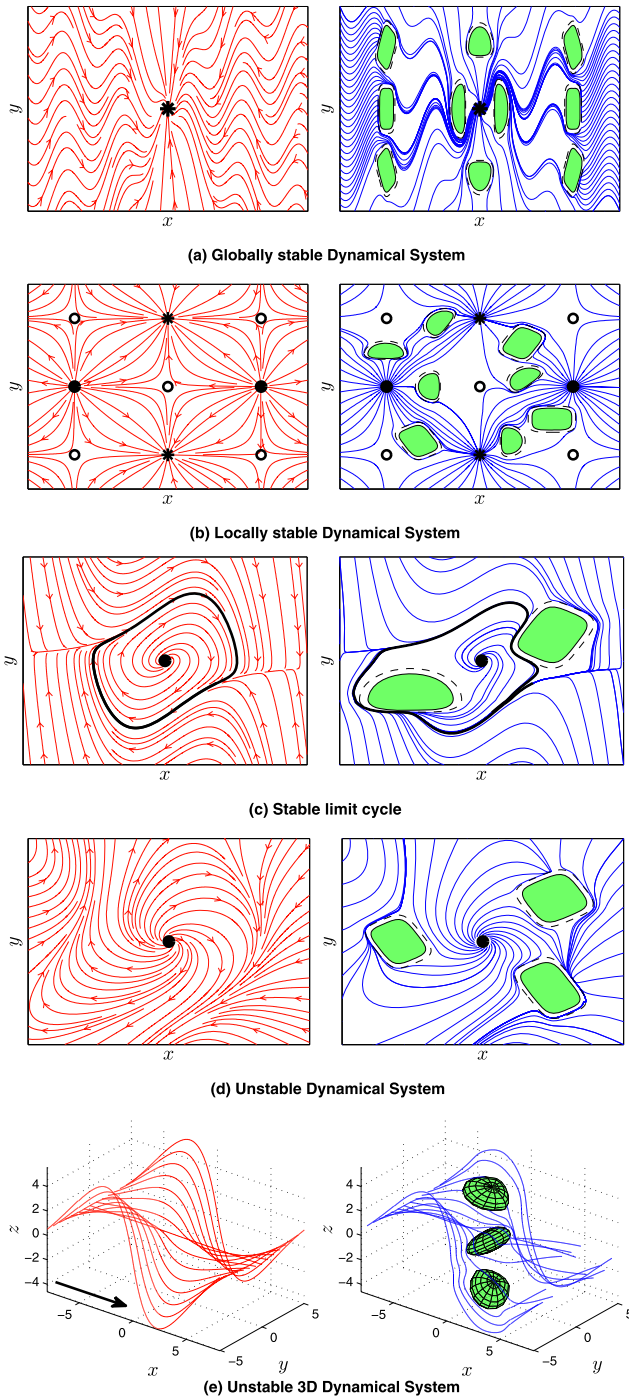


Fig. 12 Performance evaluation of the proposed obstacle avoidance module in the presence of five complex DS that are (**a**) globally stable, (**b**) locally stable, (**c**) stable limit cycle, and (**d**)–(**e**) unstable. The *left column* shows the original DS, and the *right column* illustrates the modulated DS in the presence of multiple obstacles. In this figure, stable, unstable, and saddle points are shown in *star*, *solid circle* and *hollow circle*, respectively. Obstacles are colored in *green* and the *black dashed lines* illustrate their safety margin ($\eta = 1.2$ is considered for all the obstacles). In (**c**), the *thick black line* is the stable limit cycle. For formulation of the DS and the obstacles please refer to the text in Sect. 8.1

The third DS has a stable limit cycle that includes an unstable point located at origin. The forth DS is globally unstable and has a unique unstable point at the origin. Due to the sine terms, this DS also displays a high nonlinear behavior. The fifth DS is globally unstable without equilibrium point.

All these DS are evaluated in the presence of multiple obstacles. For simplicity, we consider two types of the 2D obstacles and one 3D obstacle, but we use them in different scales, orientations, and reference points. These obstacles are formulated as follows:

$$\text{Obstacle \#1 }: \Gamma(\tilde{\xi}) = (\tilde{x}/0.75)^4 + (\tilde{y}/1)^2 = 1$$

$$\text{Obstacle \#2 }: \Gamma(\tilde{\xi}) = \begin{cases} (\tilde{x}/1.2)^4 + (\tilde{y}/0.4)^2 = 1 & y \leq y^o \\ (\tilde{x}/1.2)^2 + (\tilde{y}/1)^2 = 1 & y > y^o \end{cases}$$

$$\text{Obstacle \#3 }: \Gamma(\tilde{\xi}) = \begin{cases} \tilde{x}^2 + (\tilde{y}/1.4)^2 + (2\tilde{z})^2 = 1 & y \leq y^o \\ \tilde{x}^2 + \tilde{y}^4 + (2\tilde{z})^2 = 1 & y > y^o \end{cases}$$

Considering Fig. 12, all obstacles can be successfully avoided in all types of DS even in the presence of high nonlinearities and/or having several equilibrium points. As it is expected, the multiplication of the combined dynamic modulation matrix does not modify the original equilibrium points of the system, and does not add any extra equilibrium point in the free space $\tilde{\mathscr{X}}^f$. The potential spurious equilibrium points on the boundaries of obstacles are also handled using Algorithm 1.

8.2 Simulation experiments on SEDS/DMP

In this section we evaluate the performance of the proposed approach to generate handwritten trajectories forming the alphabet letters 'N', 'G' and 'J'. Each motion was demonstrated three times. They were collected at 50 Hz from pen input using a Tablet-PC. The motions are learned using SEDS and DMP. SEDS builds an estimate of the motion through an autonomous DS $\dot{\xi} = f(\xi)$, and thus in the presence of obstacle(s) it can be modulated by following Eq. (19), whereas DMP models a motion as a second order DS that takes the form of $\ddot{\xi} = g(t, \xi, \dot{\xi})$. This function can be transformed into a first order DS via:

$$\begin{cases} \dot{\xi} = \zeta \\ \dot{\zeta} = g(t, \xi, \zeta) \end{cases} \tag{28}$$

and the modulation due to the presence of obstacle(s) can be obtained as follows:[8]

$$\begin{cases} \dot{\xi} = M(\tilde{\xi})\zeta \\ \dot{\zeta} = g(t, \xi, M(\tilde{\xi})\zeta) \end{cases} \tag{29}$$

[8]The same principle can be used if the SEDS motions are modeled with a second or higher order DS.

Figure 13 illustrates the results for these motions in the presence of four different obstacles. In this experiment the obstacles are modeled with the following formulations:

$$\text{(a) } \Gamma(\tilde{\xi}) : \begin{cases} (\tilde{x}/20)^2 + (\tilde{y}/10)^2 = 1 & x \leq x^o \\ (\tilde{x}/20)^6 + (\tilde{y}/10)^2 = 1 & x > x^o \end{cases}$$

$$\text{(b) } \Gamma(\tilde{\xi}) : \begin{cases} (\tilde{x}/12)^2 + (\tilde{y}/1.6)^2 = 1 & x \leq x^o,\ y \leq y^o \\ (\tilde{x}/32)^2 + (\tilde{y}/1.6)^2 = 1 & x > x^o,\ y \leq y^o \\ (\tilde{x}/32)^2 + (\tilde{y}/5.6)^2 = 1 & x > x^o,\ y > y^o \\ (\tilde{x}/12)^2 + (\tilde{y}/5.6)^2 = 1 & x \leq x^o,\ y > y^o \end{cases}$$

$$\text{(c) } \Gamma(\tilde{\xi}) : \begin{cases} (\tilde{x}/12)^4 + (\tilde{y}/4)^2 = 1 & y \leq y^o \\ (\tilde{x}/12)^2 + (\tilde{y}/10)^2 = 1 & y > y^o \end{cases}$$

(d) Superposition of (a), (b), and (c)

The obstacles in Figs. 13(a) and (b) are rotated by 110° and 10°, respectively. We used the safety factor $\eta = 1.3$ for all the obstacle models. For both autonomous and non-autonomous DS, the modified dynamics of the motions successfully reach the target without hitting the obstacles. Figure 13(d) shows the result for the case where multiple objects exist in the experiment.

8.3 Robot experiments

In this section we evaluate our obstacle avoidance method in five robot experiments (three in the Cartesian space and two in the robot joint space) performed on 7 DoF Barrett WAM arm. The arm length is 1.1 m (when fully stretched). Depending on the experiment, the robot is kinematically controlled in either Cartesian or joint space, and in all cases the controller command is sent at 500 Hz. For the experiments in the Cartesian space, we use the damped least square pseudo-inverse kinematics to compute the robot's joint angles. The torque command to the robot is computed based on the desired kinematic command using the WAM built-in PID controller. All the results illustrated in this section were recorded from the robot. Recordings of the robot experiments are provided in Online Resource 1.

8.3.1 Experiments in the Cartesian space

The first experiment consisted of having the robot reach for an object while avoid hitting a table and a box. The height, length, and width of the table are 0.02, 3, and 3 m respectively, and for the box these values are 0.24, 0.36, and 0.12 m. Note that we consider an extremely large value for the length and width of the table to limit all trajectories to the region above the table. The orientation and the position of the box are computed by detecting the four markers' location (blobs) placed on the box at the rate of 100 fps us-

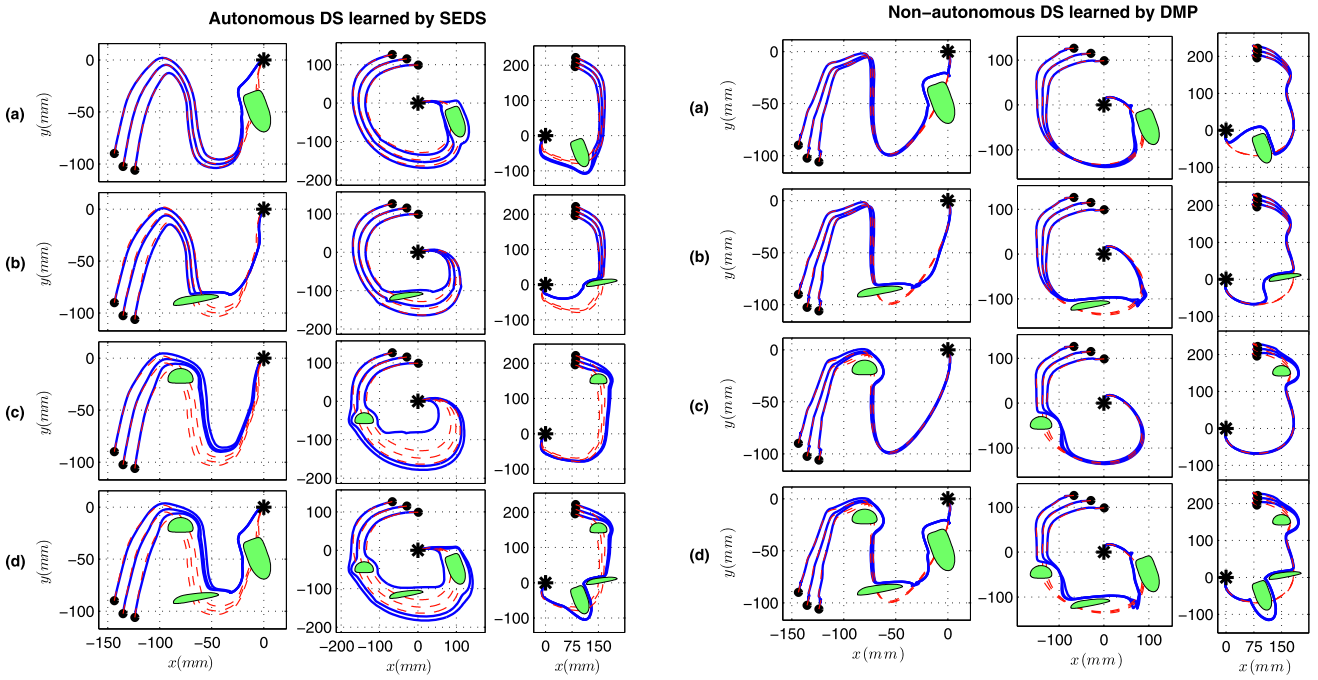
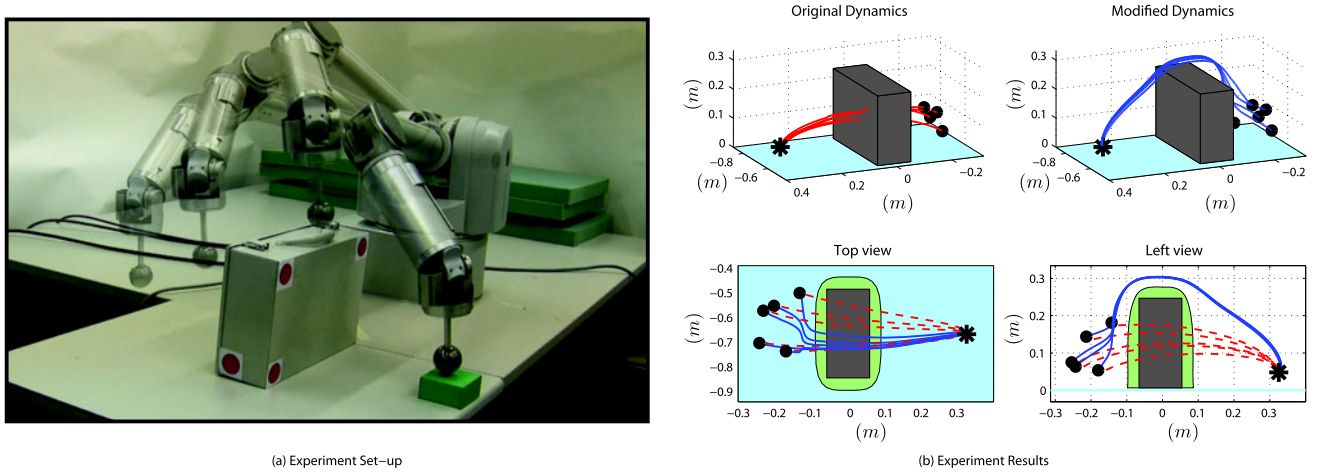


Fig. 13 Performance evaluation of the proposed approach on following three patterns in the presence of different obstacles. The motion patterns are modeled with two different approaches: (*left*) Autonomous DS using SEDS learning algorithm and (*right*) Non-autonomous DS using DMP. In the initial and final points of the trajectories are indicated by *solid circle* and *star*, respectively. Please refer to the text for further information



(a) Experiment Set–up

(b) Experiment Results

Fig. 14 (**a**) The experiment set-up. The upper surface of the *green block* corresponds to the target point. (**b**) Adaptation of the original dynamics of the reaching motion (*top-left*) with the dynamic modulation matrix (*top-right*). The graphs in the *bottom row* illustrate the top and left views of both dynamics. *Red dashed line* and *solid blue lines* correspond to the trajectories from the original and the modified dynamics, respectively. The *black area* represents the box outer surface, and the *green area* is its estimated analytical model. The *light blue rectangle* shows the upper surface of the table. The initial and final points of each trajectory are indicated by *solid circle* and *star*, respectively

ing two hi-speed Mikrotron MK-1311 cameras. The position and orientation of the table are fixed and are given to the system.

In this experiment we define the motion in the Cartesian coordinates system. The original robot motion is learned using SEDS based on a set of demonstrations (in the absence of obstacles) provided by the user. Figure 14 represents the experiment set-up and the trajectories generated from the original and the modulated dynamics of the motion. As it is expected, all reproductions from the modified dynamics

successfully avoid the box and reach the target. In this experiment, the box center is initially placed at $x^{c,B} = 0.0$, $y^{c,B} = -0.65$, and $z^{c,B} = 0.135$ with respect to the robot frame of reference. We define the box reference point to be at $x^{o,B} = x^{c,B}$, $y^{o,B} = y^{c,B}$, and $z^{o,B} = 0$, and use the analytical formulation $\Gamma(\tilde{\xi})^B$: $((x - x^{o,B})/0.092)^4 + ((y - y^{o,B})/0.23)^4 + ((z - z^{o,B})/0.27)^4 = 1$ to model the box. The table is also modeled with $x^{o,T} = y^{o,T} = 0$, $z^{o,T} = -0.01$ cm and $\Gamma(\tilde{\xi})^T$: $((x - x^{o,T})/3)^6 + ((y - y^{o,T})/3)^6 + ((z - z^{o,T})/0.01)^4 = 1$. We set the safety factor of the table to $\eta = 1.3$. For the box, we used three different values for the safety factor, i.e. $\eta_x = 2.5$, $\eta_y = 1.5$, and $\eta_z = 1.2$, to account for the large differences between the box height, length, and width.

Note that, though the box and the table are connected, we can avoid the problem highlighted in Fig. 8(d) by defining $z^{o,B} = 0$. In this way, the dynamic modulation matrix of the box always deforms trajectories towards its upper part. Thus no local minimum can be generated at the contact edges of the box and the table.

Adaptation to change in the target position To verify the adaptability of the system in a dynamic environment, we perform an experiment in which we continuously displace the target while the robot approaches it (see Fig. 15). During the reproduction, the position of the target is updated based on the output of the stereo vision system. Since the modulated dynamics preserves the asymptotic stability of the model, the system can adapt its motion on-the-fly to the change in the target position. Note that the instant adaptation to the target position is an inherent property of the SEDS modeling. In this experiment we are demonstrating the fact that our approach preserves all the properties of the SEDS model, while enabling it to perform obstacle avoidance.

Adaptation to change in both the target and obstacle positions To evaluate the performance of the system in the presence of a moving obstacle, we extend the previous example to a case where both the target and the obstacle positions are changed as the robot approaches the target. Please note that in this experiment we assume that the obstacle movement is "quasi-static". This assumption requires the obstacle approaching speed (the projection of the obstacle velocity onto the vector connecting the obstacle center to the robot end-effector) to be significantly smaller than the robot movement in that direction. Figure 16 demonstrates the obtained results. In this experiment, at the time between $t = 0$ and $t = 6$ seconds, the target is moved from its original position first in the opposite and then along the direction of the y-axis. The box also starts moving in the period between $t = 0$ and $t = 2$ seconds. During the reproduction, the target position and the box center and orientation are continuously updated based on the output of the stereo vision system. Similarly to the previous example, the system remains robust to these changes in the environment and successfully reaches the target.

Evaluation in a more dynamic environment We further evaluate our approach in a more dynamic environment where both the target and the obstacle are quickly displaced as the robot moves toward the target. Both positions of the target and the obstacle are detected at 100 Hz. The obstacle is a ball with radius 5 cm. We set its safety factor to $\eta = 1.5$. Note that the safety factor of 1.5 results in a 2.5 cm safety margin around the ball which is necessary to compensate for the size of the haptic ball attached to the robot end-effector. Figure 17 shows the experiment set-up and the obtained results. The robot adapts on-the-fly its motion to both the obstacle and the target movement.

Evaluation in a complex environment In this experiment we evaluate our method in the presence of several obstacles including a desk lamp, a pile of books, a Wall-E toy, a pencil sharpener, a book, a (red) glass, and a desk. The task consists of having the robot place a (transparent) glass on the desk, and in front of the person (see Fig. 18). The position and orientation of all the objects except the glass are pre-set. In order to have a more realistic experiment, at each trial we add a error vector ε to the predefined position of each obstacle $\xi^{o,i}$ to account for uncertainty in the environment, i.e. $\hat{\xi}^{o,i} = \xi^{o,i} + \varepsilon^i$. The value of each component of the error vector ϵ^i is drawn from a Gaussian distribution with $\mathcal{N}(0, 0025)$. The position of the glass is actively tracked through the stereo camera described above. The maximum tracking error in sensing the glass position is ± 0.05 m. The orientation of the glass is not measured, though it may change during each trial. We approximate all the obstacles with an ellipsoid envelope of the form $\sum_{i=1}^{3} (\tilde{\xi}_i / a_i)^{2p_i} = 1$, where $a_i > 0$ and $p_i > 0$ are real and integer values, respectively. To compensate for the uncertainties, we consider a safety factor of $\eta = 1.5$ for all the obstacles. The tail-effect of all the obstacles is removed (i.e. $\kappa = 0$), and the reactivity to the presence of the glass is increased by setting $\rho = 2$ (the default value of $\rho = 1$ is considered for other objects).

In this paper, we report on two trials of this experiment, but we have also included two additional trials in the accompanying video. We use the same DS function that was described in the previous robot experiments to control the robot motions. In the first trial, the person moves the red glass from his right to his left hand side (i.e. along the negative direction of the y-axis) while the robot is approaching the target point. The person intentionally moves the glass in a way that crosses the robot trajectory to the target point (see Fig. 18(a)). In order to avoid hitting the red glass, the robot deflects its trajectory towards the negative direction of y-axis, and then approaches the target from its left side (in Fig. 18(b), see the robot trajectory along y-axis in the time period $t = [3\ 4]$ seconds).

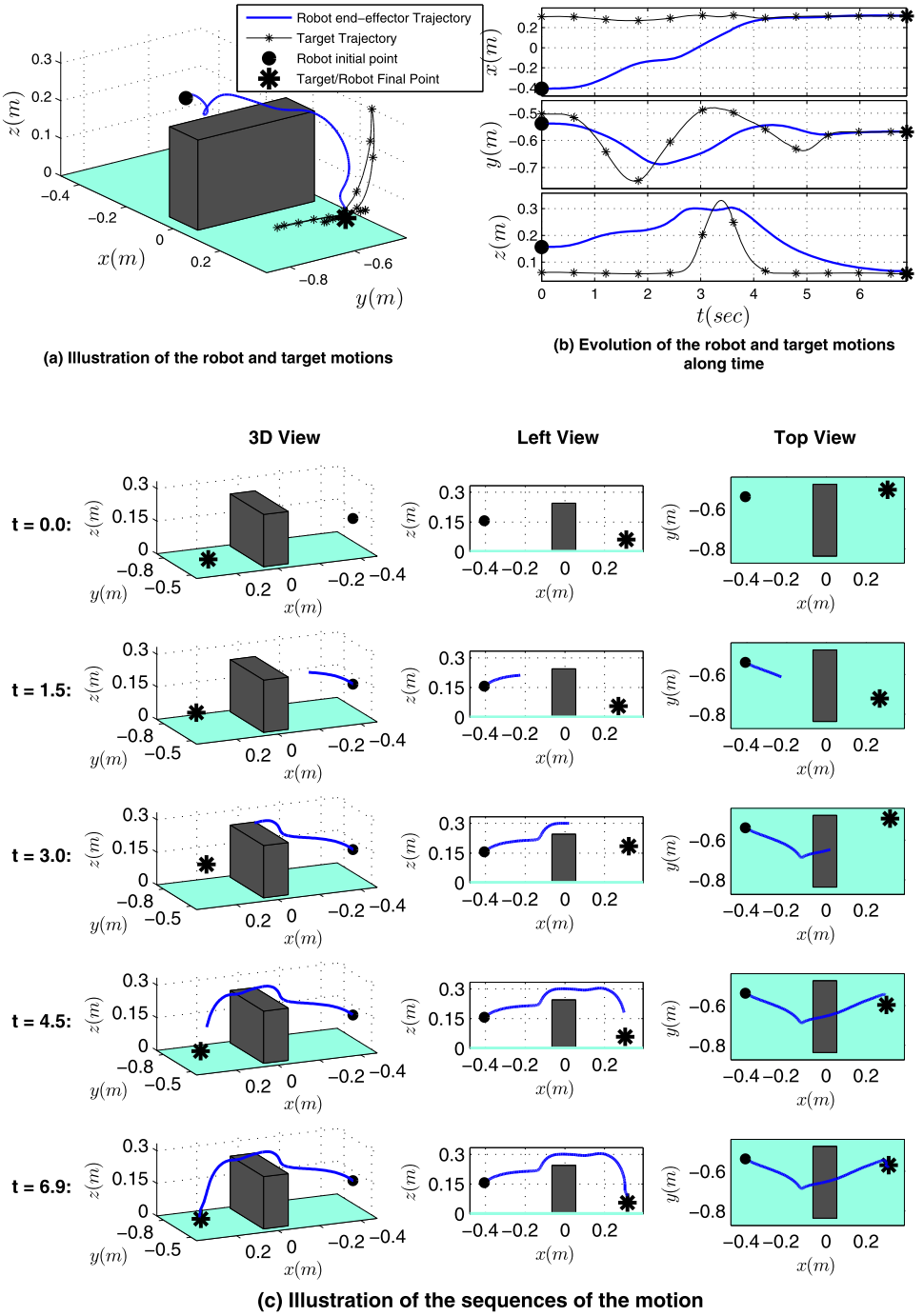
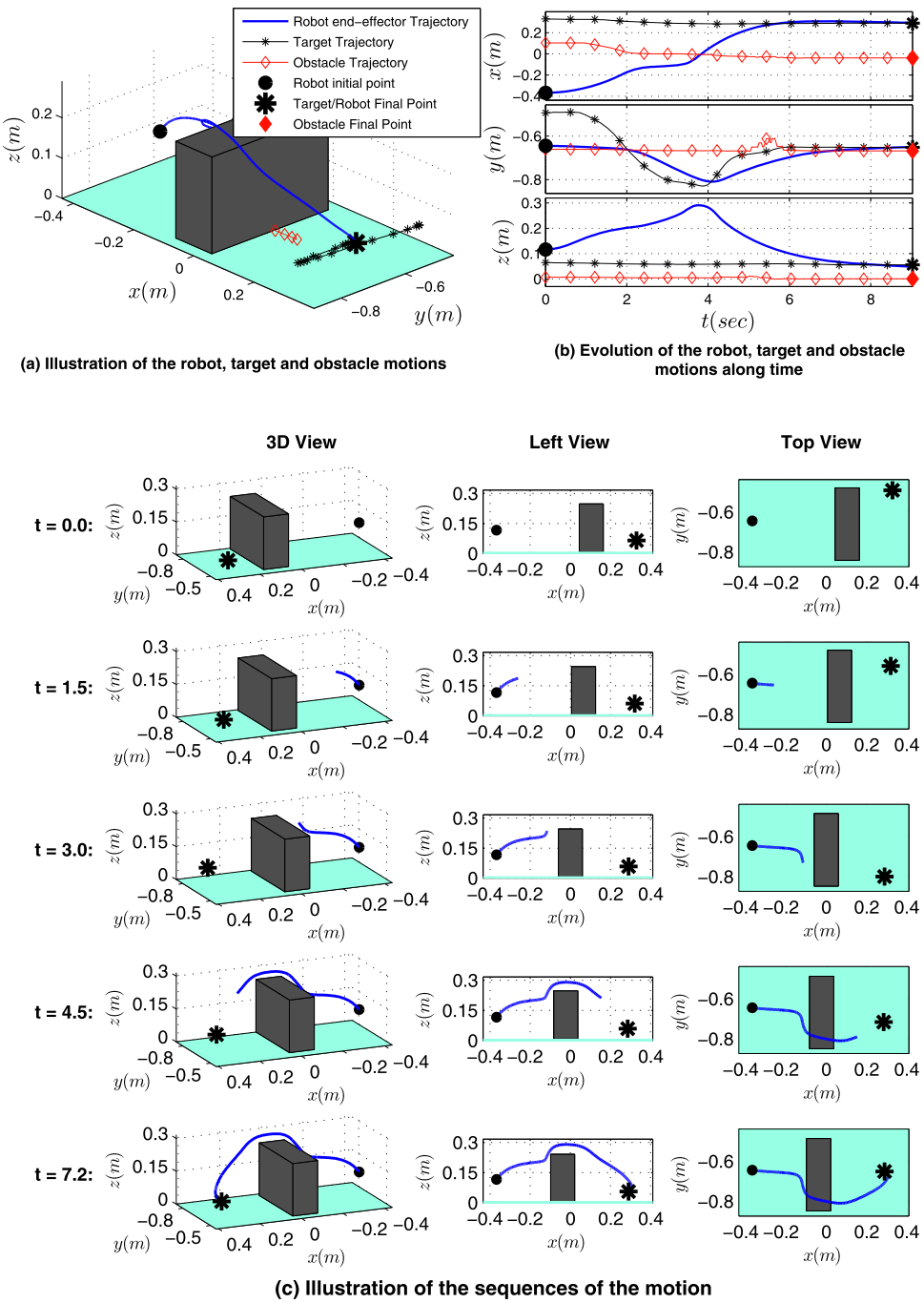


Fig. 15 Adaptation of the model to the changes in the target position

In the second trial, the person takes the glass from its right hand side and moves it to the target position while the robot is approaching. In this situation, the robot stops near the red glass (and the target) since it cannot get any closer to the target (in Fig. 18(d), see the time evolution of the robot trajectory in the time period $t = [4\ 6]$ seconds). The robot waits at this position until the person clears the areas. When the red glass is lifted, the robot moves towards the target point.

8.3.2 Experiments in the joint space

In this section, we validate our approach in $d = 7$ dimensions, by controlling this time the WAM arm's 7 joints, i.e. $\xi = [\theta_i]$, $i = 1..d$. In the first experiment, we use our obstacle avoidance approach to limit the movement range in the second joint of the robot to values below -1.2 radian. To reach this goal, we define a 7-dimensional obstacle $\Gamma(\theta) = \sum_{i=1}^{7}((\theta_i - \theta_i^o)/a_i)^4$ with $a_i = [10; 0.1; 10; 10; 10; 10; 10]$,



(a) Illustration of the robot, target and obstacle motions

(b) Evolution of the robot, target and obstacle motions along time

(c) Illustration of the sequences of the motion

Fig. 16 Robustness of the model to the changes in the target and obstacle positions

$\theta^o = [0; -1.1; 0; 0; 0; 0; 0]$ and the safety factor $\eta = 1.2$. The original DS is defined in the joint space and is learned based on a set of demonstrations in the robot joint space using the SEDS learning algorithm. Figure 19 illustrates the generated trajectories from the original and the modified dynamics. As it is expected, in the modified dynamics, the robot successfully reaches the target while the value of the second joint remains below the desired value.

In the last experiment, we use our approach to avoid two 7D spherical obstacles defined in the robot joint space. The original robot motion is a cyclic movement in θ_1-θ_2 plane with $\dot{\theta}_1 = \theta_2$ and $\dot{\theta}_2 = -\theta_1 + \theta_2(1 - (\theta_1/5)^2)$ and $\dot{\theta}_i = 0$, $\forall i \in 3..7$. The obstacles have radius of $r^{o,1} = r^{o,2} = 5$ degrees and are placed in $\theta^{o,1} = [-100; 45; 1; 61; 1; -29; 1]$ and $\theta^{o,2} = [-80; 45; -1; 59; -1; -31; -1]$, respectively. The safety factor of $\eta = 1.2$ is used in this experiment.

Figure 20(a) illustrates the evolution of the motion in the absence and presence of the obstacles. One can observe that the modulated dynamics deviates in the presence of obstacles, and due to the induced coupling via the dynamic

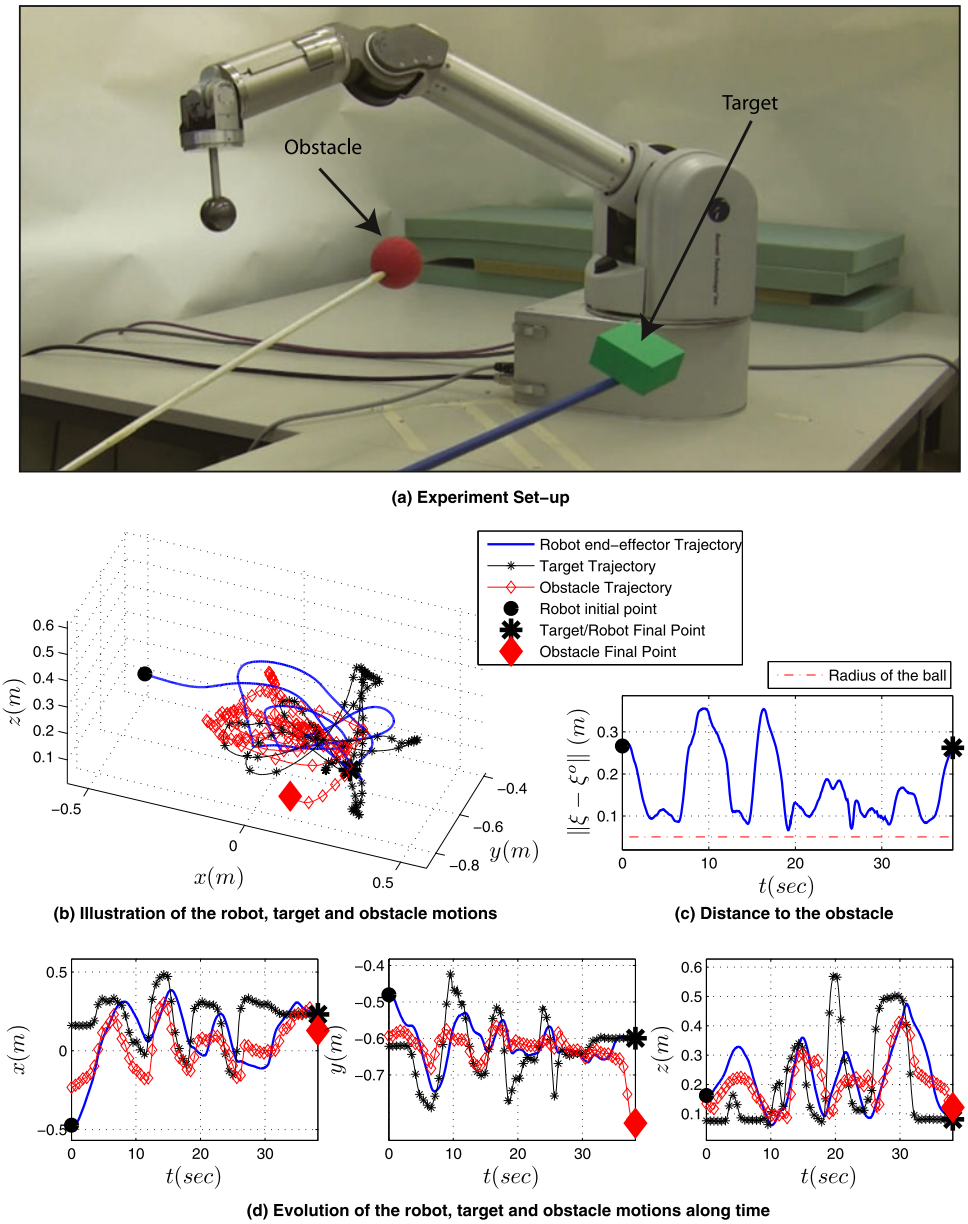


Fig. 17 Validation of the proposed method in a dynamic environment, where both the target and the obstacle are displaced continuously. The obstacle is a ball with the radius of 5 cm. Please refer to the text for the further information

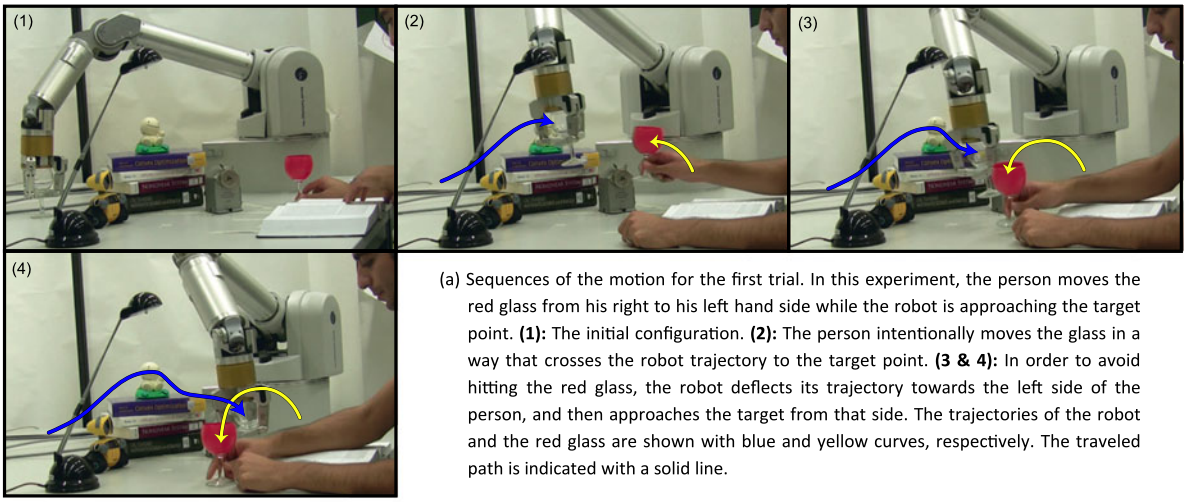
modulation matrix,[9] the robot also starts showing cyclic behavior in previously static joints, i.e. θ_i, $i = 3..7$. Figure 20(b) shows the distance to the closest obstacle along the time. Here, one can observe that while the original motion penetrates into the obstacle, the modulated dynamics can smoothly avoid the obstacles. The evolution of the motion along time is shown in Fig. 20(c). One can see that the period of the motion is slightly decreased due to the presence of the obstacle.[10] The corresponding robot motion in the task space is shown in Fig. 20(d).

9 Summary and conclusion

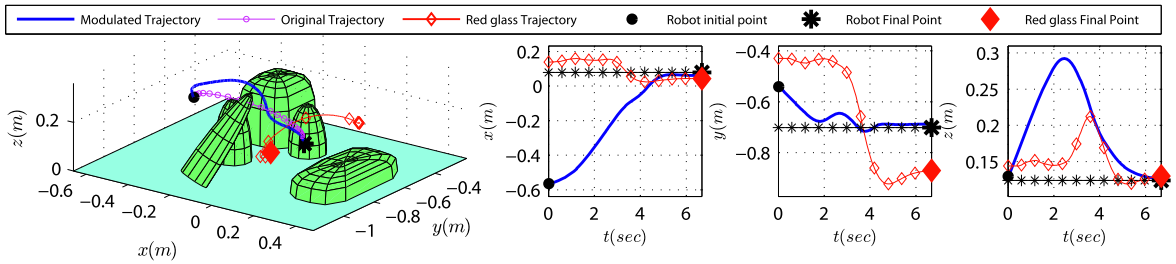
In this paper, we proposed a Dynamical System approach to realtime obstacle avoidance for a case where robot motions are given by autonomous or non-autonomous DS, and

[9]Note that the motions across θ_i, $i = 3..7$ would become uncoupled if the obstacles were placed at $\theta^{o,1} = [-100; 45; 0; 60; 0; -30; 0]$ and $\theta^{o,2} = [-80; 45; 0; 60; 0; -30; 0]$.

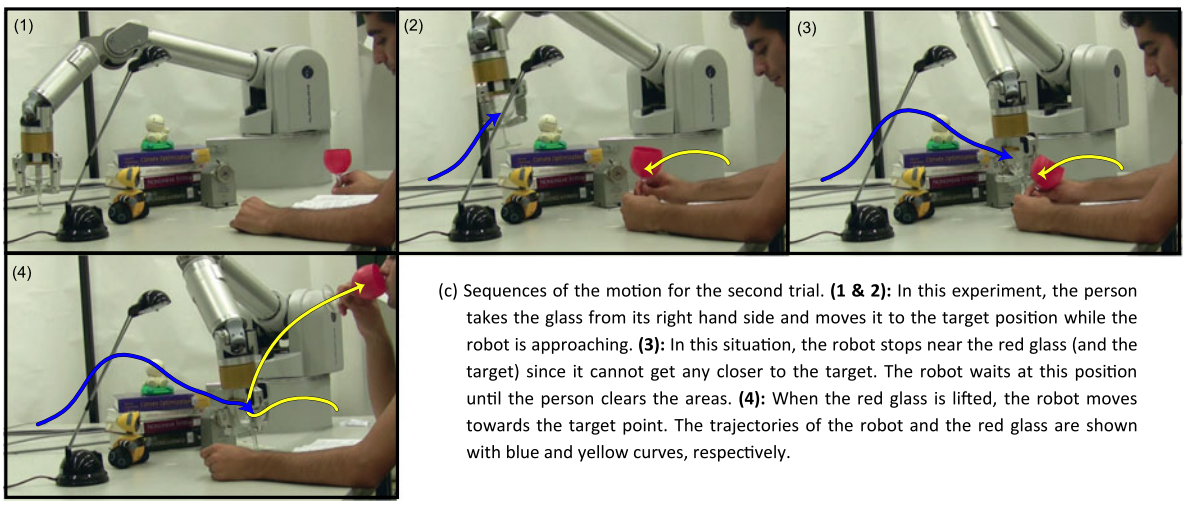
[10]Note that this paper does not claim that the cyclic behavior is always preserved in the presence of the obstacles.



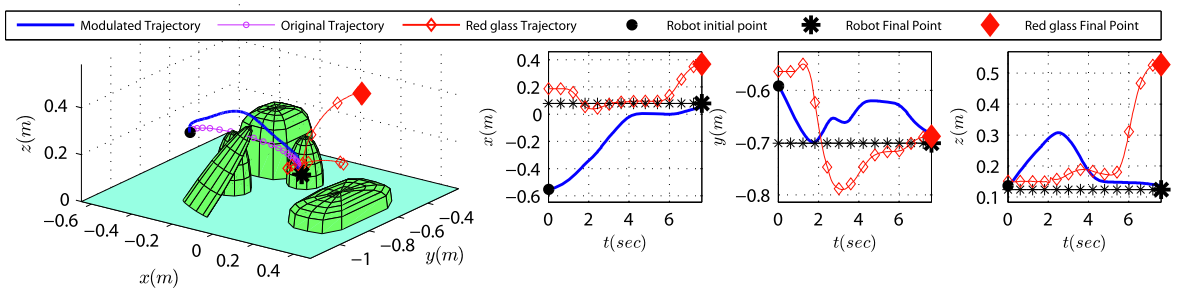
(a) Sequences of the motion for the first trial. In this experiment, the person moves the red glass from his right to his left hand side while the robot is approaching the target point. **(1):** The initial configuration. **(2):** The person intentionally moves the glass in a way that crosses the robot trajectory to the target point. **(3 & 4):** In order to avoid hitting the red glass, the robot deflects its trajectory towards the left side of the person, and then approaches the target from that side. The trajectories of the robot and the red glass are shown with blue and yellow curves, respectively. The traveled path is indicated with a solid line.



(b) Illustration of the robot, target and obstacle motions for the first trial.



(c) Sequences of the motion for the second trial. **(1 & 2):** In this experiment, the person takes the glass from its right hand side and moves it to the target position while the robot is approaching. **(3):** In this situation, the robot stops near the red glass (and the target) since it cannot get any closer to the target. The robot waits at this position until the person clears the areas. **(4):** When the red glass is lifted, the robot moves towards the target point. The trajectories of the robot and the red glass are shown with blue and yellow curves, respectively.



(d) Illustration of the robot, target and obstacle motions for the second trial.

Fig. 18 Evaluation of the proposed method in a complex environment. In this experiment, the robot is required to put a glass on the desk and in front of the person, while avoid hitting several objects including a desk lamp, a pile of books, a Wall-E toy, a pencil sharpener, an open book, a (*red*) glass, and a desk. All the objects except the *red glass* are fixed and their convex envelope are shown in *green*. The trajectory of the *red glass* is indicated by *red diamonds* (for the clarity of the graph, we do not display the envelope of the *red glass*)

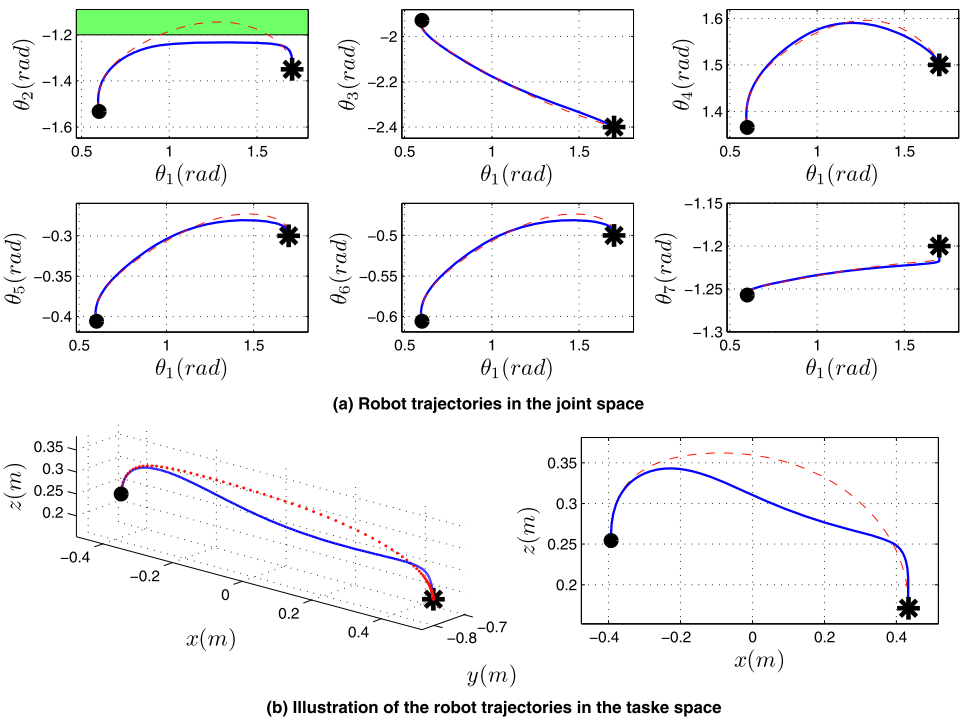


Fig. 19 Using the proposed obstacle avoidance module to limit the movement range in the second joint of the robot to values below −1.2 radian. (**a**) The *red dashed line* and the *blue solid line* corresponds to the trajectories generated by the original and the modified dynamics, respectively. The obstacle is shown in *green*. The initial and final points of the motion are indicated by *solid circle* and *star*, respectively. (**b**) Illustration of the robot movement in the robot task space

the obstacle(s) are convex. The method is derived for a d-dimensional DS, hence can be used in both the Cartesian and configuration spaces. The proposed method can handle multiple obstacles, and do not modify the equilibrium points of the original dynamics. However, in the presence of obstacle(s) the method may lead to the appearance of saddle points and local minima along the obstacles' boundaries. These points can be tackled through Algorithm 1.

The presented approach requires a global model of the environment and an analytical modeling of the obstacles boundary. When the analytical description of the obstacle is available, our method guarantees that all obstacles will be avoided safely. However, the analytical equation of the obstacle or its accurate status (i.e. position and orientation) may not be available all the time. To generate the analytical equation, it is possible to use one of the state-of-the-art bounding-volume algorithms (e.g. Benallegue et al. 2009; Lahanas et al. 2000; Welzl 1991) to approximate a convex BV on the output of the vision system. In this work we used the approach by Benallegue et al. (2009) because it satisfies the convexity and $C1$ smoothness conditions required in our approach, and it provides a good volume-ratio convex fit of objects. In the worst case when there is little time to generate the bounding volume, one could quickly fit the point cloud with an ellipsoid.

The presented algorithm is able to cope with uncertainty in the obstacle's position by allowing certain safety margins around the obstacle. The larger the safety margin, the more robust the system is to uncertainty in the obstacle position. Note that in the presence of an unforeseen object or uncertainty in the obstacle's position, our algorithm no longer guarantees the safe avoidance of the obstacle, and can only strive for the best performance.

All theorems derived in this work are based on the continuous state space assumption; however, in real experiments, robot motions are usually generated with a finite number of points (discrete modeling). Thus the choice of integration time step is important specially in the close vicinity of the object. In fact, when a big integration time step is used, for trajectories that are very close to an obstacle, it is very likely that the subsequent point falls inside the obstacle due to the integration error. In this situation, trajectories tend to remain inside the obstacle (because the boundaries are impenetrable, no trajectory can enter or leave the obstacle). In this paper, we did not face such an issue by considering the integration time steps of 0.01 and 0.002 sec in all simulations and robot experiments, respectively.

The presented work is limited in that it can only be applied to convex shaped obstacles. While Theorem 2 still holds for concave shape, the simple algorithm I to overcome local minima on the boundary can no longer apply and an alternative solution must be sought. As a part of future work, we are aiming at developing a non-harmonic formulation of the panel method to model concave obstacles.

The quasi-static assumption that is considered in this paper for moving obstacles is quite conservative. An important extension to this work is to relax this assumption. Such ex-

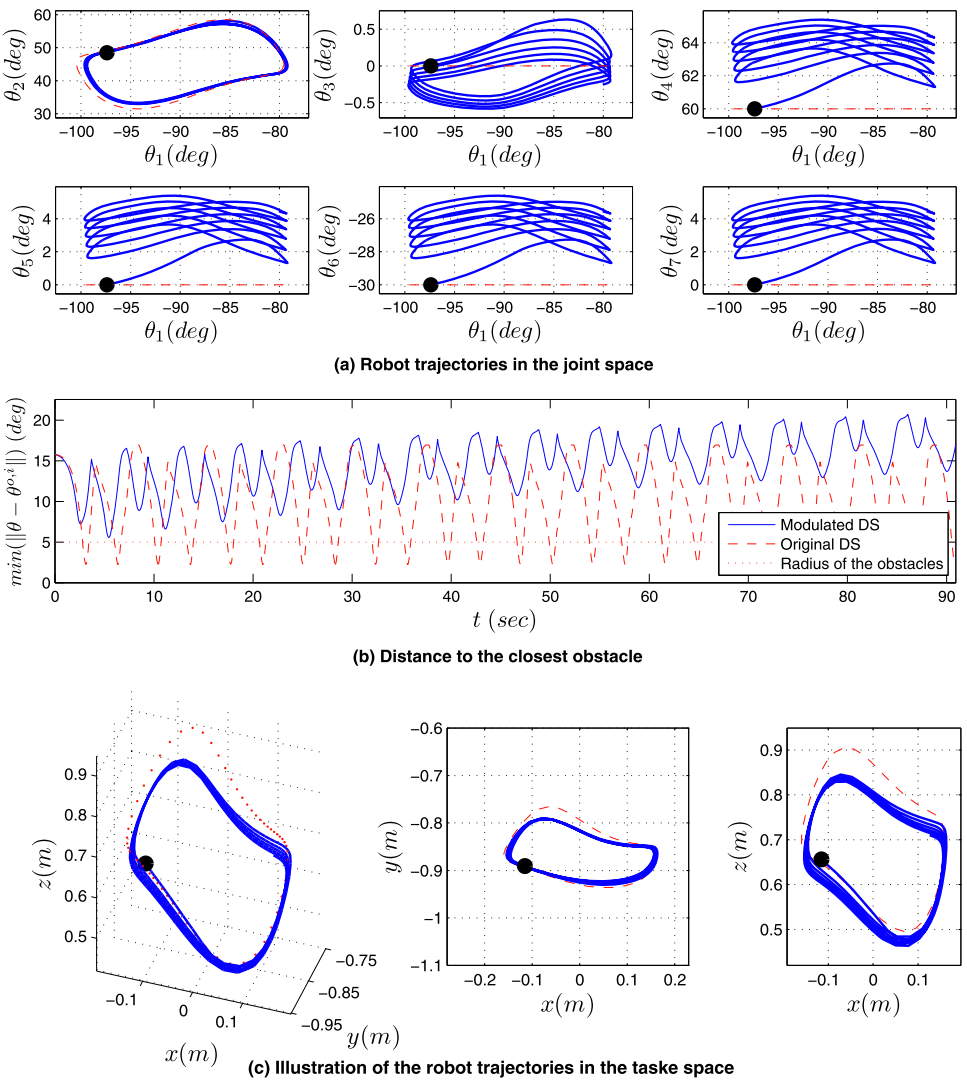


Fig. 20 Illustration of applying the obstacle avoidance module in the robot joint space. In this figure, the red dashed line shows the original cyclic motion and the *solid line* demonstrates the modulated motion in the presence of two 7-D spherical obstacles with the radius of 10 degrees. The robot motion is defined in the joint space and its evolution is shown in (**a**). The *solid black circle* indicates the starting point of the motion. The distance to the closest obstacle is illustrated in (**b**). The corresponding robot motion in the task space is shown in (**c**). Please refer to the text for the further information

tension currently exists for the case where the robot and the obstacle motions are defined by harmonic functions (Feder and Slotine 1997). However, further investigation should be carried out for non-harmonic motions.

The presented work considers obstacle avoidance for a point robot. However, it is also possible to integrate other algorithms to perform collision avoidance for the whole robot. For example, while the end-effector follows the commanded velocity from the proposed approach, one can use the kinematics null-space to avoid link collision (Maciejewski and Klein 1985). Furthermore, similarly to Park et al. (2008), we could also use the presented approach to control the kinematics null-space movement. To do this, it is only necessary to find the closest point on the robot to the obstacle, and then use the proposed model to drive away this point from the obstacle (if this movement is feasible in the joint null-space).

The source code of the proposed obstacle avoidance module can be downloaded from: http://lasa.epfl.ch/sourcecode/.

Acknowledgements This work was supported by the European Commission through the EU Project AMARSI (FP7-ICT-248311). The authors kindly thank E. Sauser for providing the RobotToolkit interface to control the Barrett WAM arm, and M. Duvanel for the vision system. The authors also thank M. Benallegue and A. Kheddar for providing the source code of the STP-BV method to generate bounding volumes from the point cloud of objects.

Appendix A: Proof of Theorem 1

Consider a hyper-surface $\mathscr{X}^b \subset \mathbb{R}^d$ corresponding to boundary points of a hyper-sphere obstacle in $\mathbb{R}^d$ with a center ξ^o and a radius r^o. Impenetrability of the obstacle's boundaries is ensured if the normal velocity at boundary points $\xi^b \in \mathscr{X}^b$ vanishes:

$$n\left(\xi^b\right)^T \dot{\xi}^b = 0 \quad \forall \xi^b \in \mathscr{X}^b \tag{30}$$

where $n(\xi^b)$ is the unit normal vector at a boundary point ξ^b:

$$n\big(\xi^b\big) = \frac{\xi^b - \xi^o}{\|\xi^b - \xi^o\|} \xrightarrow{\tilde{\xi}^b = \xi^b - \xi^o} n\big(\xi^b\big) = \frac{\tilde{\xi}^b}{r} \quad \forall \xi^b \in \mathscr{X}^b \tag{31}$$

The eigenvalue decomposition of the square matrix $M^s(\tilde{\xi}, r^o)$ is given by:

$$M^s\big(\tilde{\xi}, r^o\big) = V^s\big(\tilde{\xi}, r^o\big)\, D^s\big(\tilde{\xi}, r^o\big)\, V^s\big(\tilde{\xi}, r^o\big)^{(-1)} \tag{32}$$

where $D^s(\tilde{\xi}, r^o)$ is a $d \times d$ diagonal matrix composed of the eigenvalues:

$$\begin{cases} \lambda^1 = 1 - \frac{r^2}{\tilde{\xi}^T \tilde{\xi}} \\ \lambda^i = 1 + \frac{r^2}{\tilde{\xi}^T \tilde{\xi}} \quad \forall i \in 2..d \end{cases} \tag{33}$$

and $V^s(\tilde{\xi}, r^o) = [\upsilon^1 \ \cdots \ \upsilon^d]$ is the matrix of eigenvectors with:

$$\begin{cases} \upsilon^1 = \tilde{\xi} \\ \upsilon^i_j = \begin{cases} -\tilde{\xi}_i & j = 1 \\ \tilde{\xi}_1 & j = i \\ 0 & j \neq 1, i \end{cases} \quad \forall i \in 2..d,\ j \in 1..d \end{cases} \tag{34}$$

Substituting Eqs. (31), (32) and (7) into Eq. (30) yields:

$$n\big(\xi^b\big)^T \dot{\xi}^b = \frac{(\tilde{\xi}^b)^T}{r} V^s\big(\tilde{\xi}^b, r^o\big)\, D^s\big(\tilde{\xi}^b, r^o\big)\, V^s\big(\tilde{\xi}^b, r^o\big)^{(-1)} f(.) \tag{35}$$

Since ξ^b is equal to the first eigenvector of $V^s(\tilde{\xi}^b, r^o)$, Eq. (35) reduces to:

$$n\big(\xi^b\big)^T \dot{\xi}^b = \begin{bmatrix} r \\ [\mathbf{0}]_{d-1} \end{bmatrix}^T D^s\big(\tilde{\xi}^b, r^o\big)\, V^s\big(\tilde{\xi}^b, r^o\big)^{(-1)} f(.) \tag{36}$$

where $[\mathbf{0}]_{d-1}$ is a zero column vector of dimension $d-1$. For all points on the obstacle boundary, the first eigenvalue is zero, i.e. $\lambda^1 = 0$, $\forall \xi^b \in \mathscr{X}^b$. Thus, we have:

$$n\big(\xi^b\big)^T \dot{\xi}^b = [\mathbf{0}]_d^T\, V^s\big(\tilde{\xi}^b, r^o\big)^{(-1)} f(.) = 0 \tag{37}$$

Appendix B: Proof of Theorem 2

The proof of Theorem 2 follows directly from that of Theorem 1. Observe that:

$$n\big(\xi^b\big)^T \dot{\xi}^b = n\big(\xi^b\big) E\big(\tilde{\xi}^b, r^o\big)\, D\big(\tilde{\xi}^b, r^o\big)\, E\big(\tilde{\xi}^b, r^o\big)^{(-1)} f(.) \tag{38}$$

Considering the fact that $n(\xi^b)$ is equal to the first eigenvector of $E(\tilde{\xi}^b, r^o)$, and the first eigenvalue is zero for all points on the obstacle boundary yields:

$$\begin{aligned} n\big(\xi^b\big)^T \dot{\xi}^b &= \begin{bmatrix} 1 \\ [\mathbf{0}]_{d-1} \end{bmatrix}^T D\big(\tilde{\xi}^b, r^o\big)\, E\big(\tilde{\xi}^b, r^o\big)^{(-1)} f(.) \\ &= [\mathbf{0}]_d^T\, E\big(\tilde{\xi}^b, r^o\big)^{(-1)} f(.) = 0 \end{aligned} \tag{39}$$

References

Barbehenn, M., Chen, P. C., & Hutchinson, S. (1994). An efficient hybrid planner in changing environments. In *IEEE int. conf. on robotics and automation* (Vol. 4, pp. 2755–2760).

Benallegue, M., Escande, A., Miossec, S., & Kheddar, A. (2009). Fast C1 proximity queries using support mapping of sphere-torus-patches bounding volumes. In *Proc. IEEE int. conf. on robotics and automation* (pp. 483–488).

Borenstein, J., & Koren, Y. (1991). The vector field histogram—fast obstacle avoidance for mobile robots. *IEEE Transactions on Robotics and Automation*, *7*, 278–288.

Brock, O., & Khatib, O. (2002). Elastic strips: A framework for motion generation in human environments. *The International Journal of Robotics Research*, *21*(12), 1031–1052.

Burns, B., & Brock, O. (2005). Toward optimal configuration space sampling. In *Proc. of robotics: science and systems*.

Diankov, R., & Kuffner, J. (2007). Randomized statistical path planning. In *Proc. of IEEE/RSJ int. conf. on robots and systems (IROS)* (pp. 1–6).

Feder, H. J. S., & Slotine, J.-J. E. (1997). Real-time path planning using harmonic potentials in dynamic environments. In *Proc. of IEEE int. conf. on robotics and automation (ICRA)* (Vol. 1, pp. 874–881).

Fraichard, T., Hassoun, M., & Laugier, C. (1991). Reactive motion planning in a dynamic world. In *Proc. of the IEEE int. conf. on advanced robotics* (pp. 1028–1032).

Hoffmann, H., Pastor, P., Park, D.-H., & Schaal, S. (2009). Biologically-inspired dynamical systems for movement generation: automatic real-time goal adaptation and obstacle avoidance. In *Proc. of int. conf. on robotics and automation* (pp. 2587–2592).

Iossifidis, I., & Schöner, G. (2006). Dynamical systems approach for the autonomous avoidance of obstacles and joint-limits for a redundant robot arm. In *Proc. of the IEEE int. conf. on intelligent robots and systems (IROS)* (pp. 580–585).

Kavraki, L. E., Svestka, P., Latombe, J.-C., & Overmars, M. H. (1996). Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation*, *12*(4), 566–580.

Khansari-Zadeh, S. M., & Billard, A. (2011). Learning stable non-linear dynamical systems with Gaussian mixture models. *IEEE Transactions on Robotics*, *27*(5), 943–957. ISSN 1552-3098. doi:10.1109/TRO.2011.2159412.

Khatib, O. (1986). Real-time obstacle avoidance for manipulators and mobile robots. *The International Journal of Robotics Research*, *5*, 90–98.

Kim, J.-O., & Khosla, P. K. (1992). Real-time obstacle avoidance using harmonic potential functions. *IEEE Transactions on Robotics and Automation*, *8*(3), 338–349.

Kuffner, J. J., & LaValle, S. M. (2000). RRT-connect: An efficient approach to single-query path planning. In *Proc. of IEEE int. conf. on robotics and automation* (Vol. 2, pp. 995–1001).

Lahanas, M., Kemmerer, T., Milickovic, N., Karouzakis, K., Baltas, D., & Zamboglou, N. (2000). Optimized bounding boxes for

three-dimensional treatment planning in brachytherapy. *Medical Physics*, *27*(10), 2333–2342.

Lozano-Perez, T. (1983). Spatial planning: A configuration space approach. *IEEE Transactions on Computers*, *30*, 108–120.

Lumelsky, V., & Skewis, T. (1990). Incorporating range sensing in the robot navigation function. *IEEE Transactions on Systems, Man, and Cybernetics*, *20*, 1058–1069.

Maciejewski, A. A., & Klein, C. A. (1985). Obstacle avoidance for kinematically redundant manipulators in dynamically varying environments. *The International Journal of Robotics Research*, *4*, 109–116.

Milne-Thomson, L. M. (1960). *Theoretical hydrodynamics* (4th edn.) London: Macmillan.

Park, D.-H., Hoffmann, H., Pastor, P., & Schaal, S. (2008). Movement reproduction and obstacle avoidance with dynamic movement primitives and potential fields. In *Proc. of the IEEE int. conf. on humanoid robotics* (pp. 91–98).

Quinlan, S., & Khatib, O. (1993). Elastic bands: connecting path planning and control. In *Proc. of the IEEE int. conf. on robotics and automation (ICRA)* (Vol. 2, pp. 802–807).

Shilane, Ph., Min, P., Kazhdan, M., & Funkhouser, Th. (2004). The Princeton shape benchmark. In *Shape modeling international*, Italy.

Simmons, R. (1996). The curvature-velocity method for local obstacle avoidance. In *Proc. of the IEEE int. conf. on robotics and automation* (Vol. 4, pp. 3375–3382).

Sprunk, Ch., Lau, B., Pfaffz, P., & Burgard, W. (2011). Online generation of kinodynamic trajectories for non-circular omnidirectional robots. In *Proc. of IEEE int. conf. on robotics and automation (ICRA)* (pp. 72–77).

Toussaint, M. (2009). Robot trajectory optimization using approximate inference. In *25th int. conf. on machine learning (ICML)* (pp. 1049–1056).

Vannoy, J., & Xiao, J. (2008). Real-time adaptive motion planning (ramp) of mobile manipulators in dynamic environments with unforeseen changes. *IEEE Transactions on Robotics*, *24*, 1199–1212.

Waydo, S., & Murray, R. M. (2003). Vehicle motion planning using stream functions. In *Proc. of the IEEE int. conf. on the robotics and automation (ICRA)* (Vol. 2, pp. 2484–2491).

Welzl, E. (1991). Smallest enclosing disks (balls and ellipsoids). In H. Maurer (Ed.), *New results and new trends in computer science* (Vol. 555, pp. 359–370). Berlin/Heidelberg: Springer.

Yang, Y., & Brock, O. (2007). Elastic roadmaps: Globally task-consistent motion for autonomous mobile manipulation in dynamic environments. In *Proc. robotics: science and systems*.

Yoshida, E., & Kanehiro, F. (2011). Reactive robot motion using path replanning and deformation. In *Proc. IEEE int. conf. on robotics and automation* (pp. 5457–5462).

Seyed Mohammad Khansari-Zadeh is Ph.D. student in Robotics at the school of Engineering at EPFL. He received his M.Sc. (2008) in Flight Dynamics and Control from department of Aerospace Engineering at Sharif University of Technology. His research interests include modeling, control, and stability analysis of nonlinear dynamics and using imitation learning to infer control policies for robot control.

Aude Billard is Associate Professor and head of the Learning Algorithms and Systems Laboratory at the school of Engineering at EPFL. She received her B.Sc. (1994) and M.Sc. (1995) in Physics from EPFL and her Ph.D. in Artificial Intelligence (1998) from the University of Edinburgh.