Real-time Reactive Motion Generation Based on Variable Attractor Dynamics and Shaped Velocities

Sami Haddadin, Holger Urbanek, Sven Parusel, Darius Burschka, Jürgen Roßmann,
Alin Albu-Schäffer, and Gerd Hirzinger

Abstract— **This paper describes a novel method for motion generation and reactive collision avoidance. The algorithm performs arbitrary desired velocity profiles in absence of external disturbances and reacts if virtual or physical contact is made in a unified fashion with a clear physically interpretable behavior. The method uses physical analogies for defining attractor dynamics in order to generate smooth paths even in presence of virtual and physical objects. The proposed algorithm can, due to its low complexity, run in the inner most control loop of the robot, which is absolutely crucial for safe Human Robot Interaction. The method is thought as the locally reactive real-time motion generator connecting control, collision detection and reaction, and global path planning.**

I. INTRODUCTION

Future robots will work closely with humans in industrial environments, which necessitate safe robot design [1], sophisticated control methods for realizing soft robotics features [2], and collision detection algorithms with appropriate reaction strategies [6], [7]. Furthermore, it is of major importance to provide flexible motion generation methods, which take into account the possibly complex environment structure and at the same time can react very quickly to changing conditions.

Motion generation methods can be divided into path planning algorithms and reactive motion generation. On the one hand (probabilistic) complete, highly sophisticated offline path planning methods are used, which provide complete collision free paths for potentially complex scenarios [4] with multi degree-of-freedom (DoF) open or closed chain kinematics. On the other hand, reactive motion generators, which usually provide a more responsive behavior, are of simpler character and have very short execution cycles. Both classes mostly treat the entire motion generation problem from a purely geometric/kinematic point of view. Unfortunately, both methods have significant drawbacks.

With the recent advances in physical Human-Robot Interaction (pHRI) it becomes even more important to be able to plan complex motions for task achievement **and** cope with the proximity of dynamic obstacles under the absolute premise of safety to the human at the same time. Complex motion planners cannot match the real-time requirements of the low level control cycle due to their computational complexity. Reactive methods on the other hand do usually not provide completeness and are prone to get stuck in local minima. This necessitates to treat motion planning, collision avoidance, and collision detection/reaction not separated anymore. Of course, global planning methods have to generate some valid path for the coarse motion of the robot, but we believe absolute optimality and absolute collision avoidance have not the highest priority in highly dynamic environments, since the overall execution time, robustness, and flexible reaction are of higher interest. In order to satisfy the requirements posed by very quick and safe reaction cycles, real-time methods have to be used for local motions that can fully exploit the capabilities of the robot.

However, it is not satisfactory anymore to only circumvent objects while preventing contact. Contact has to be an integral part of the reactive motion scheme since it could be the vital part of the task. In any case, we believe the role of collisions should be redefined, since absolute avoidance is artificially restricting robots with high performance sensing capabilities that are designed for handling contact. With such devices collisions with the environment do not have to be avoided by all means as long as they do not create harmful situations to humans or the environment or conflict with the task. Therefore, contact force information should be integrated into the collision avoidance schemes so that in case unexpected contact occurs, e.g. due to incomplete/inconsistent knowledge of the environment or unpredicted behavior of the human, the robot can retract and circumvent the sources of external forces in a similar fashion to virtual forces that are e.g. generated via potential fields. In other words a much more flexible trajectory deformation during contact is desirable, which does not shift the entire load of coping with the collision to the control schemes, but actively responds to unexpected events. Furthermore, a common problem with reactive strategies is their non predictable behavior in case of virtual/physical external forces. Especially in human environments, it cannot be an option to avoid an upcoming object very quickly by increasing unpredictably velocity into another direction and then possibly collide with another object or human.

In this paper we present a new real-time method for reactive collision avoidance that can also cope with external forces and furthermore is able to serve as a general purpose interpolator with arbitrary desire velocity profile. Even in case of external contacts, we provide a clear behavior of the robot and use the information of contact for deforming the trajectory safely in real time. The accompanying video showcases the performance of the method.

This paper is organized as follows. Section II gives a short summary on the state of the art of reactive motion generation, followed by the design concept of the proposed algorithm in Sec. III with some simple simulation to illustrate the idea. Sec. IV presents the experimental performance of the

S. Haddadin, H. Urbanek, S. Parusel, A. Albu-Schäffer, and G. Hirzinger are with Institute of Robotics and Mechatronics, DLR - German Aerospace Center, Wessling, Germany sami.haddadin, holger.urbanek, sven.parusel, alin.albu-schaeffer@dlr.de

Darius Burschka is with Lab for Robotics and Embedded Systems at Technische Universität München, Garching, Germany and Institute for Robotics and Mechatronics, DLR - German Aerospace Center, Wessling, Germany burschka@cs.tum.de

Jürgen Roßmann is with Institute of Man-Machine Interaction at Rheinisch-Westfälische Technische Hochschule Aachen, Aachen, Germany rossmann@mmi.rwth-aachen.de

proposed method for static and dynamic obstacles. Finally we conclude in Sec. V.

II. State-of-the-art

Path planning with reactive collision avoidance was mostly investigated in the field of mobile manipulators [19], [11]. A typical task to be fulfilled is to avoid obstacles with or without (partial) task consistency, which are either known beforehand or suddenly appear, thus necessitating quick response times.

For real-time collision avoidance the potential field based methods are powerful schemes [15], [8], [19], [11]. A virtual repulsive potential is assigned to each known obstacle and an attractive potential to the desired goal configuration. This leads to a directed motion towards the goal while avoiding the obstacles in a reactive fashion. In [11], e.g., the method is applied for the translative motion of a mobile base and in [19] for a manipulator mounted on a mobile base alone. Furthermore, the potential fields can be extended from a virtual point-shaped particle model to various shapes for the robot. These are able to change their orientation accordingly to avoid obstacles [10]. Despite one of its major deficits, namely its possibility to get easily stuck in local minima, its very fast calculation time within the low-level controller cycle of the robot is a very well known benefit.

One possibility to overcome this drawback is presented with the circulatory fields, introduced in [16]. Each obstacle is attached with a circulatory field, similar to that of an electrical charge in a magnetic field. While this field will then drive the path around the obstacle, this method will not be able to find optimal solutions. However, in principle this method is free of local minima.

A concept proposed in [13], [14], which is closely related to potential fields is **CARE**. Based on proximity and relative velocity it generates evasive joint accelerations for avoiding external objects. The method is straight forward to be used for multi-robot systems as well.

Other promising principles of combining a global path planner with a local collision avoidance strategy are the Elastic Strips [3] or the preceeding Elastic Bands [12]. A global path planner searches for a path around the known obstacles. Unforseen appearing hindrance can then deform the planned path as if it was a rubber band, while avoidance of known obstacles is still possible. The elastic strips and elastic bands, however, are computationally more complex, so that they can hardly run in the inner most control loop, especially for full multi-DoF robots. Therefore, they increase the time lag until a reaction to an obstacle initiates.

Instead of applying potential field methods in the Cartesian space, one could also apply them in the configuration space (C-space) of the robot [18]. Still, since calculations in the C-space (especially for a large number of DoFs) are computationally complex, this method is practically only applicable for offline planning, and therefore offers no reactive behavior. This method, however, is able to find valid paths, where Cartesian space based potential field methods fail.

A somewhat different approach, which focus was to increase safety is given in [5]. Given a collision free path, a so called proxy acts as an attractor, which slides along the path, yet having its own dynamics, therefore smoothing out discontinuities of the given path. The robot is then connected to the proxy by a PID like controller. This combination allows for a safer, gently path-following robot, avoiding sudden jerky motions.

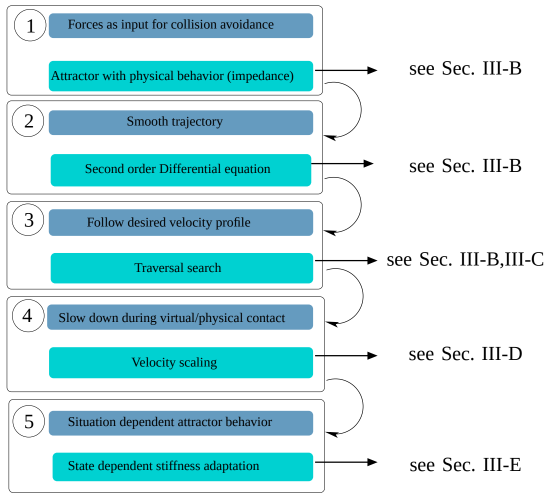

1
Forces as input for collision avoidance
Attractor with physical behavior (impedance)
see Sec. III-B
2
Smooth trajectory
Second order Differential equation
see Sec. III-B
3
Follow desired velocity profile
Traversal search
see Sec. III-B,III-C
4
Slow down during virtual/physical contact
Velocity scaling
see Sec. III-D
5
Situation dependent attractor behavior
State dependent stiffness adaptation
see Sec. III-E

Fig. 1. Design steps for the proposed algorithm.

A very good general overview of classical motion planning techniques for reactive planning is given in [9], where also work on discrete potential fields is reviewed.

After this brief overview on reactive collision avoidance methods we illustrate our concept of a collision avoidance system, which provides solutions to some of the aforementioned limitations of existing methods.

III. Algorthm design

A. Main idea of the method

The collision avoidance technique presented in this paper is based on the attractor idea of the potential field method. We provide significant extensions, which help to overcome some of its major drawbacks. Figure 1 shows the consecutive desired behaviors (①-⑤), visualizing the design process of the algorithm. In addition, the proposed schemes we chose to fulfill our requirements are given and the according sections referred to.

First of all, we seek for a real-time collision avoidance method that can run in the inner most control loop of the robot (in our case at 1 kHz). Furthermore, we want virtual and physical forces to be the input for avoiding collisions or retract from them ①. Therefore, we chose an impedance equation and selected a decoupled second order differential equation for sake of smoothness of the generated motion ②. In order to be able to follow arbitrary desired velocity profiles, we traverse the predicted path of the resulting attractor dynamics every time step. Then, we choose the configuration along this trajectory that matches the associated desired velocity value ③. This enables us to use only the geometric properties of the calculated path, having the nice characteristics of the attractor, while forcing the motion generator to produce the commanded desired velocities along this path.

Especially during physical contact it is often desirable to slow down motion, which we ensure by velocity scaling ④. Finally, we alter the coupling to the goal (the attractor stiffness) depending on the current state ⑤. This can be interpreted as a temporal detachment from the goal configuration during (virtual) contact. After this process the coupling is restored again, leading to the continuation of goal convergence. This prevents the unnecessary fighting between attractive and repulsive forces.

B. Attractor design

Potential Field methods as introduced in [8] are well known for their computational efficiency and general applicability. Thus, they have become a standard method in robotics [15]. In the original work a potential field was introduced that consists of a driving attractor for reaching the target configuration, while the robot is being deviated form its desired motion by virtual objects that generate repelling virtual forces. Formally, it can be described by

$$\begin{aligned}\mathbf{f}(\mathbf{x}_d, \mathbf{x}_d^*, \mathbf{x}_o) &= \mathbf{f}_a(\mathbf{x}_d, \mathbf{x}_d^*) + \mathbf{f}_r(\mathbf{x}_d, \mathbf{x}_o) \\ &= \mathbf{f}_a(\mathbf{x}_d) + \sum_k \mathbf{f}_{r_k}(\mathbf{x}_d, \mathbf{x}_{o_k}),\end{aligned} \tag{1}$$

with $\mathbf{x}_d, \mathbf{x}_d^*, \mathbf{x}_{o_k} \in \mathbb{R}^n$ being the position of the virtual particle, the desired goal configuration and the closest point of the Surface $\mathbb{S}_k$ of the kth repulsive object. $\mathbf{f} : \mathbb{R}^n \times \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^n$, $\mathbf{f}_a, \mathbf{f}_r : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^n$ are the resulting driving, attractive, and repulsive forces associated with the potential field $V : \mathbb{R}^n \times \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^n$ via

$$\mathbf{f}(\mathbf{x}_d, \mathbf{x}_d^*, \mathbf{x}_o) = -\frac{\partial V(\mathbf{x}_d, \mathbf{x}_d^*, \mathbf{x}_o)}{\partial \mathbf{x}_d}. \tag{2}$$

The resulting repulsive force usually consists of the sum of the k repulsive components $\mathbf{f}_{r_k} : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^n$. The attractive force is expressed by the first order differential equation

$$\mathbf{f}_a(\mathbf{x}_d) = K_v\,(\mathbf{x}_d - \mathbf{x}_d^*) + D_v \dot{\mathbf{x}}_d, \tag{3}$$

with $K_v = \operatorname{diag}\{K_{v,i}\} \in \mathbb{R}^{n \times n}, \quad i = 1 \dots n$ being a diagonal stiffness matrix and $D_v = \operatorname{diag}\{D_{v,i}\} \in \mathbb{R}^{n \times n}, \quad i = 1 \dots n$ the diagonal damping matrix. In order to bound the resulting velocity, which could in principle become very high, [8] proposed to set bounds on the desired velocity based on the norm of the desired velocity vector. This makes it possible to travel at constant maximum velocity after acceleration and before deceleration phase.

In most cases the repulsive forces are expressed as a function of the distances from the virtual particle to the repulsive elements. These objects are often chosen to be of simple geometric shape as e.g. spheres, cylinders, or planes. In order to limit their influence and provide smooth force responses, we chose a cosine-shaped blending function.

$$\mathbf{f}_{r_k}(\mathbf{x}_d, \mathbf{x}_{o_k}) = \begin{cases} \frac{(\mathbf{x}_{o_k} - \mathbf{x}_d)}{d_k} \frac{\cos\left(\frac{d_k}{d_{\max_k}}\pi\right)+1}{2} f_{\max_k} & \text{if } d_k \in [0 \dots d_{\max_k}], \\ 0 & \text{otherwise}, \end{cases} \tag{4}$$

with $d_k = \|\mathbf{x}_d - \mathbf{x}_{o_k}\|$ and $d_{\max_k}$ being the maximum distance of influence of a repulsive element. $f_{\max_k}$ is the maximum repelling force of the kth repulsive element.

For the ease of use, we omit $\mathbf{x}_d, \mathbf{x}_d^*, \mathbf{x}_o$ from now on in the force functions (using e.g. $\mathbf{f}$ instead of $\mathbf{f}(\mathbf{x}_d, \mathbf{x}_d^*, \mathbf{x}_o)$).

Apart from the slight redefinition of virtual external forces, we assign a realistic mass and inertia to the virtual particle, producing a trajectory that could in principle take into account the robot inertial properties into the commanded motion. The resulting particle dynamics are therefore defined by a second order mass-spring-damper-system.

$$M_v \ddot{\mathbf{x}}_d + K_v\,(\mathbf{x}_d - \mathbf{x}_d^*) + D_v \dot{\mathbf{x}}_d = \mathbf{f}_r, \tag{5}$$

with $M_v \in \mathbb{R}^{n \times n}$ being the virtual mass matrix.

As mentioned earlier it is important to incorporate real physical forces into the avoidance scheme to provide a more general disturbance response. Therefore, we use the real external forces $\mathbf{f}_{\text{ext}} \in \mathbb{R}^n$ that act along the robot structure as well (in combination with $K_{\text{ext}} = \operatorname{diag}\{\mathrm{K}_{\text{ext}}^{\mathrm{i}}\}, K_{\text{ext}}^i > 0$). Equation (5) becomes

$$\mathbf{f}_{r_{\text{total}}} = \mathbf{f}_r + K_{\text{ext}} \mathbf{f}_{\text{ext}} = M_v \ddot{\mathbf{x}}_d + K_v\,(\mathbf{x}_d - \mathbf{x}_d^*) + D_v \dot{\mathbf{x}}_d. \tag{6}$$

These forces are e.g. provided by a force sensor in the robot wrist or by an accurate estimation by an observer. In case of the DLR Lightweight Robot III (LWR-III), this is realized by a nonlinear disturbance observer unit for estimating the external joint torques $\boldsymbol{\tau}_{\mathbf{ext}} \in \mathbb{R}^n$. Its output is the first order filtered version of $\hat{\boldsymbol{\tau}}_{\mathbf{ext}} \in \mathbb{R}^n$, see [6], [7]. These torques can then be transformed into estimations of external forces by

$$\hat{\mathbf{f}}_{\text{ext}} = (J^T)^{\#} \hat{\boldsymbol{\tau}}_{\mathbf{ext}}, \tag{7}$$

where $J \in \mathbb{R}^{n \times m}$ is the appropriate contact Jacobian. This force estimation is now available for integration into the task space avoidance[1].

Second order differential equations as (6) are usually unsolvable for dynamic environments, producing highly nonlinear and rapidly changing virtual forces, together with basically unpredictable physical forces.

$$\mathbf{f}_{r_{\text{total}}} = \mathbf{f}_r + K_{\text{ext}} \mathbf{f}_{\text{ext}} = f(\mathbb{S}_R, \dot{\mathbb{S}}_R, \mathbb{S}_i, \dot{\mathbb{S}}_i, t, \dots) + K_{\text{ext}} \mathbf{f}_{\text{ext}} \tag{8}$$

$\mathbb{S}_R, \dot{\mathbb{S}}_R$ are the relevant surface representation of the robot and its velocity. $\mathbb{S}_i, \dot{\mathbb{S}}_i$ are the positions and velocities of static and dynamic environment objects.

Due to the mentioned induction of highly nonlinear system behavior, forward simulation of (6) needs to be used. $t_\epsilon \in \mathbb{R}^+$ is the time horizon used for calculating the desired motion.

Object motion can e.g. be given in terms of observation and prediction, so that $\mathbf{f}_r$ is representing the predicted virtual dynamics during numerical integration of (6). External forces act during one sample as a constant bias force.

Twice integration of (6) for every sample time t_n leads to the predicted path $\mathbf{m}_{\epsilon,n}(t), \; t \in [t_n \dots t_\epsilon]$ of the virtual particle:

$$\begin{aligned}\mathbf{m}_{\epsilon,n} := \mathbf{x}_d = \\ \mathbf{x}_d(t_n) + \iint_{t_n}^{t_\epsilon} M_v^{-1}\left[\mathbf{f}_{r_{\text{total}}} - K_v(\mathbf{x}_d - \mathbf{x}_d^*) - D_v \dot{\mathbf{x}}_d\right] dt + \dot{\mathbf{x}}_d(t_n)\; dt.\end{aligned} \tag{9}$$

The straight forwards choice is to set $t_\epsilon = t_n + \Delta t$ with Δt being the discrete interpolation sample time. In other words one integration step is calculated and the outcome directly used as the desired trajectory. However, such a simple solution leads for most cases to very undesired high velocities and accelerations of the generated path and therefore, does not solve the problem of most existing methods.

In order to tackle this problem, we apply the integration of (9) with a forward Euler integrator for a limited amount of $s \in \mathbb{N}^+$ steps within a certain time interval $t_\epsilon = t_n +$

[1]Of course, this estimation degrades when approaching kinematic singularities.

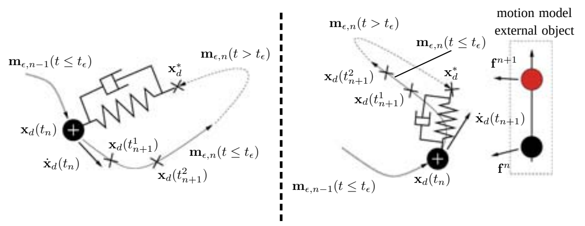

$\mathbf{m}_{\epsilon,n-1}(t \le t_\epsilon)$
$\mathbf{x}^*_d$
$\mathbf{m}_{\epsilon,n}(t > t_\epsilon)$
$\mathbf{x}_d(t_n)$
$\dot{\mathbf{x}}_d(t_n)$
$\mathbf{x}_d(t^1_{n+1})$
$\mathbf{m}_{\epsilon,n}(t \le t_\epsilon)$
$\mathbf{x}_d(t^2_{n+1})$
$\mathbf{m}_{\epsilon,n}(t > t_\epsilon)$
motion model
external object
$\mathbf{m}_{\epsilon,n}(t \le t_\epsilon)$
$\mathbf{x}_d(t^2_{n+1})$
$\mathbf{x}^*_d$
$\mathbf{f}^{n+1}$
$\mathbf{x}_d(t^1_{n+1})$
$\dot{\mathbf{x}}_d(t_{n+1})$
$\mathbf{f}^n$
$\mathbf{x}_d(t_n)$
$\mathbf{m}_{\epsilon,n-1}(t \le t_\epsilon)$

Fig. 2. Schematic views of the collision avoidance for two consecutive iteration steps. The left figure denotes free motion, whereas the right one takes into account a motion model of an external virtual object.

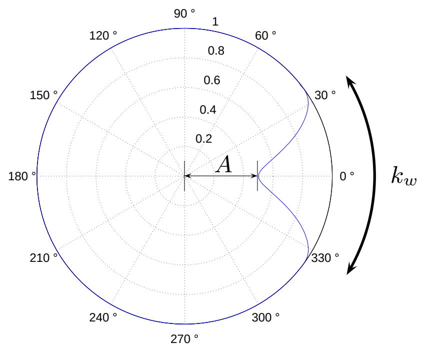


Fig. 3. Velocity scaling factor with respect to $\measuredangle(-\mathbf{f}_r, \dot{\mathbf{x}}_d)$. $0°$ means the robot is directly approaching the obstacle.

$s\Delta t$. The constant s has been chosen such that the real-time condition of the inner most control loop is not violated by the computation time. This way we traverse the predicted path of the system $\mathbf{m}_{\epsilon,n}(t' \le t_\epsilon)$ every time step, incorporating the dynamic behavior of the environment and the external forces, which are assumed to be a vector field in this prediction step. However, we dismiss the time information associated with it and instead use a new input variable, the desired track speed $\dot{x}'_d \in \mathbb{R}^+_0$. In order to match this desired velocity $\dot{x}'_d$, we search for the configuration $\mathbf{x}_d(t_{n+1})$ along the path $\mathbf{m}_{\epsilon,n}$ that ensures it.

This yields $s+1$ sampling points $\mathbf{x}_d(t^0_{n+1}) \ldots \mathbf{x}_d(t^s_{n+1})$ with the starting configuration $\mathbf{x}_d(t^0_{n+1}) = \mathbf{x}_d(t_n)$, $\dot{\mathbf{x}}_d(t^0_{n+1}) = \dot{\mathbf{x}}_d(t_n)$ being also the starting configuration of the robot. The following algorithm interpolates between the bracketing sampling points for the desired track speed $\dot{x}'_d$ and produces the according ordered configuration $\mathbf{x}_{d,\text{ord}} \in \mathbb{R}^n$.

```
$i = 0$;
$v_0 = 0$;
while $(v_i < \dot{x}'_d) \wedge (i \le s)$ do
    $i = i + 1$;
    $v_i = v_{i-1} + \frac{\|\mathbf{x}_d(t^{i-1}_{n+1}) - \mathbf{x}_d(t^i_{n+1})\|}{t^i_{n+1} - t^{i-1}_{n+1}}$;
end
if $i \le s$ then
    $\mathbf{x}_{d,\text{ord}} = \mathbf{x}_d(t^{i-1}_{n+1}) + (\mathbf{x}_d(t^i_{n+1}) - \mathbf{x}_d(t^{i-1}_{n+1}))\frac{\dot{x}'_d - v_{i-1}}{v_i - v_{i-1}}$;
end
if $i > s$ then
    $\mathbf{x}_{d,\text{ord}} = \mathbf{x}_d(t^s_{n+1})$;
end
```

If $\dot{x}'_d$ cannot be reached, because the number of integrator steps was not sufficient, the last sample point is chosen $\mathbf{x}_{\text{ord}} = \mathbf{x}_d(t^s_{n+1})$. This usually happens, if the virtual particle gets stuck in a local minimum or near the goal position $\mathbf{x}^*_d$ as the goal is asymptotically approached or $\dot{x}'_d$ was accidently commanded to jump or to be inappropriately high. A visual description of the principle is depicted in Figure 2.

To sum up, we keep the smooth properties and the inherent collision avoidance capabilities of the generated local path, but the track velocity of the robot can be commanded independently, even arbitrarily.

In the next subsections we outline the design of the different inputs and parameters of the algorithm.

C. Velocity profiles

Since the proposed method allows to use arbitrary time based input velocity profiles $\dot{x}'_d(t)$, we can realize classical trapezoidal or sinusoidal motion with inherent collision avoidance. However, time based profiles are during virtual or physical collisions only of limited use, since they are intrinsically violated when deviation from the nominal path takes place. Therefore, a distance based velocity profile is a better choice. In this paper we use the following desired velocity profile.

$$\dot{x}'_d(e_d) = \begin{cases} (v_d-\delta)\frac{1}{2}\left(1-\cos\left(\pi\left(\frac{e_d}{c_1}\right)\right)\right)+\delta & \text{if } e_d < c_1 \\ v_d & \text{if } e_d \ge c_1 \wedge e_d \le c_2 \\ (v_d-\delta)\frac{1}{2}\left(1+\cos\left(\pi\left(\frac{e_d-c_2}{1-c_2}\right)\right)\right)+\delta & \text{if } e_d > c_2 \wedge e_d < (1-\delta) \\ 0 & \text{else,} \end{cases} \tag{10}$$

where v_d denotes the nominal constant track speed and $c_1, c_2 \in \mathbb{R}^+$ the acceleration and deceleration boundaries. $\delta \in \mathbb{R}^+ \ll c_1$ is a tolerance value and $e_d \in [0 \ldots 1]$ is defined as

$$e_d := \frac{\mathbf{x}_{d,\text{ord}} - \mathbf{x}^*_{d,i}}{\|\mathbf{x}_0 - \mathbf{x}_{d,\text{ord}}\| + \|\mathbf{x}_{d,\text{ord}} - \mathbf{x}^*_{d,i}\|}. \tag{11}$$

This scalar can be interpreted as a normalized "distance to travel". This definition is chosen since it enables us to change the goal online without having to readapt the boundary values. When changing the goal from $\mathbf{x}^*_{d,1}$ to $\mathbf{x}^*_{d,2}$ during motion it is not sufficient to only use $e_d := \frac{\|\mathbf{x}_{d,\text{ord}} - \mathbf{x}^*_{d,i}\|}{\|\mathbf{x}_0 - \mathbf{x}_{d,\text{ord}}\|}$. This is due to the fact that $d_2 < d_1 + d_2$ counts while the first target is chosen but $d_3 > d_4$ when switching to the second one, potentially leading to $e_d > 1$.

D. Velocity Scaling

During (virtual) contact, the commanded velocity is, similar to the method described in [7] for physical contact, additionally shaped. Thus, due to the collision avoidance, the robot could continuously reduce speed, or even retract, and at the same time actively avoid the upcoming collision. In the most basic case the presence of external objects shall lead to an intrinsically lower velocity. For this purpose, the method of velocity scaling in case of $\|\mathbf{f}_r\| > 0$ is used to slow down the motion around objects generating these virtual forces. One extension over this pure scaling of velocities in presence of a repelling force $\|\mathbf{f}_r\| > 0$ is to scale the velocity-profile, as a function of the direction of the total repelling force $\mathbf{f}_r$ and the current motion vector $\dot{\mathbf{x}}_{d,\text{ord}}$, see Fig. 3.

1) Virtual force based: The angle between the total repelling force $-\mathbf{f}_r$ and the commanded velocity vector $\dot{\mathbf{x}}_{\text{ord}}$ is given by:

$$\phi = \arccos\left(\frac{\langle -\mathbf{f}_r, \dot{\mathbf{x}}_{\text{d,ord}}\rangle}{\|\mathbf{f}_r\|\|\dot{\mathbf{x}}_{d,\text{ord}}\|}\right), \tag{12}$$

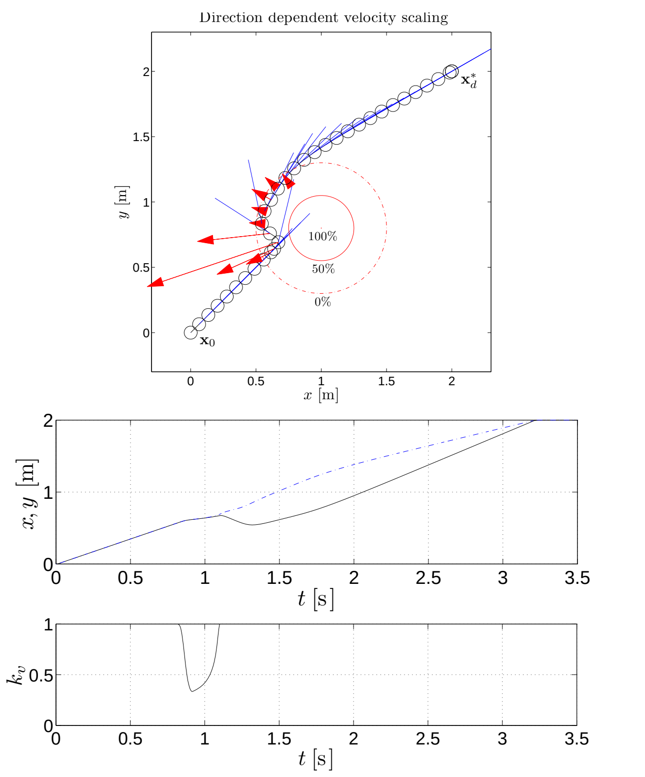


Fig. 4. Direction dependent velocity scaling depicted in 2D. The upper figure shows the position $\mathbf{x}_{d,\text{ord}}$ at equidistant time intervals of $0.1\ s$ (black obstacles). The red circles show the obstacle together with the force horizon for 0 %, 50 % and 100 %. The middle graph depicts the trajectory of the x- and y-coordinates (with x being solid and y dashed) and the lower one the velocity scaling factor k_v.

with $\phi \in [0 \ldots \pi]$. (12) is used to calculate a velocity scaling factor, given the parameter for the speed-ditch width $k_w \in [0 \ldots \pi]$ and amplitude $k_a \in [0 \ldots 1]$.

$$k_{v_{\text{virt}}}(\phi) = \begin{cases} 1 - k_a \frac{\cos\left(\frac{\phi\pi}{k_w}\right)+1}{2} & \text{if } \phi \in [-k_w \ldots k_w] \\ 1 & \text{else}, \end{cases} \tag{13}$$

where $k_{v_{\text{virt}}} \in [0 \ldots 1]$. For ensuring a smooth velocity change, $k_a \in [0 \ldots 1]$ is defined as a function of $\|\mathbf{f}_r\|$,

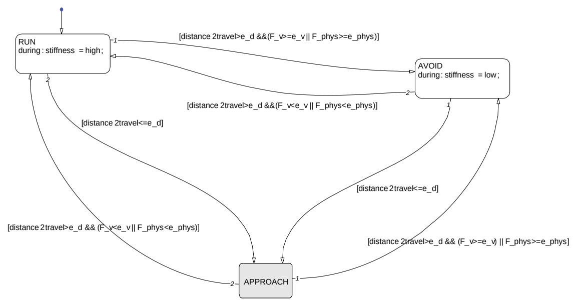
$$k_a = \begin{cases} A\left(1 - \frac{\cos\left(\frac{\|\mathbf{f}_r\|}{f_{\max}}\pi\right)+1}{2}\right) & \text{if } \|\mathbf{f}_r\| \le f_{\max}, \\ A & \text{else}, \end{cases} \tag{14}$$

$f_{\max} \in \mathbb{R}^+$ being some force saturation constant and $A \in [0 \ldots 1]$. Note, that $k_{v_{\text{virt}}}(\phi)$ is symmetric: $k_{v_{\text{virt}}}(-\phi) = k_{v_{\text{virt}}}(\phi)$. Therefore, the restriction of (12) to $[0 \ldots \pi]$ does not generate any conflict.

2) Physical force based: Scaling down the velocity can be very useful during physical contact. This is simply done by applying a monotonically decreasing scaling function g

$$k_{v_{\text{ext}}} = g(\mathbf{f}_{\text{ext}}), \tag{15}$$

with $k_{v_{\text{ext}}} \in \mathbb{R}^+$. In order to incorporate physical forces, there are various behaviors that are desirable. An intuitive choice is to slow down if motion and force vector point in different directions and to accelerate if their direction is similar.



RUN
during: stiffness = high;
AVOID
during: stiffness = low;
APPROACH
[distance 2travel>e_d &&(F_v>=e_v || F_phys>=e_phys)]
[distance 2travel>e_d &&(F_v<e_v || F_phys<e_phys)]
[distance 2travel<=e_d]
[distance 2travel<=e_d]
[distance 2travel>e_d && (F_v<e_v || F_phys<e_phys)]
[distance 2travel>e_d && (F_v>=e_v) || F_phys>=e_phys]
1
2
2
1
2
1

Fig. 5. State depending scaling of the attractor stiffness.

3) Fusion: In order to fuse both scaling factors consistently, we use the more conservative one.

$$k_v = \min(k_{v_{\text{virt}}}(\phi), k_{v_{\text{ext}}}(\mathbf{f}_{\text{ext}})), \tag{16}$$

with $k_v \in [0 \ldots 1]$. Therefore, the direction dependent desired track speed $\dot{x}''_d \in \mathbb{R}$ becomes

$$\dot{x}''_d = k_v \dot{x}'_d, \tag{17}$$

leading to a slowdown of the motion as long as the robot drives towards critical obstacles, but leaves the desired velocity untouched if bypassing or departing.

E. Stiffness adaptation

The attractor stiffness enables us to change the overall attractor behavior online according to the current situation. High stiffness relates of course to higher convergence rate, whereas decreasing values represent an increasing decoupling from the goal configuration, therefore allowing much better avoidance behavior.

We exploit the full state information $e_d, \mathbf{f}_r$, and $\mathbf{f}_{\text{ext}}$ to achieve higher performance. Figure 5 shows the overall state depending stiffness behavior of the attractor. We define the discrete states the attractor can occupy as *RUN*, *AVOID*, and *APPROACH*. If no avoidance is needed, which is the case if the goal is not approached yet, we set the diagonal stiffness values to very high values that are in the order of magnitude of the physical reflected robot stiffness and furthermore a function of e_d.

$$K_v^{\text{high}} = \max\{K_v^{\max}(1 - e_d), K_v^{\min}\} \tag{18}$$

This way we provide best possible convergence when approaching. In case avoiding behavior (due to virtual or physical forces) is desired, a relaxing behavior is activated, which enables almost decoupling of virtual particle and goal configuration. In the *APPROACH* state, a more complex behavior is necessary in case any disturbance is present. Figure 6 depicts the overall block diagram of the method.

In the next section we analyze the experimental performance of the proposed method for the fully torque controlled LWR-III in various situations with static and dynamic obstacles.

IV. Experiments

A. The DLR Lightweight Robot III

The LWR-III is a 7DoF robot with a weight of 14 kg and a nominal payload of 7 kg [2]. It is equipped with a joint torque

[2]This value is the nominal payload according to industrial long range testing. However, for research purposes the robot can carry its own weight.

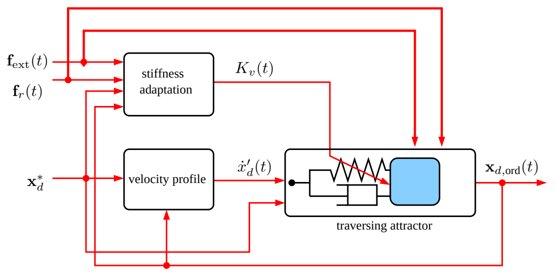


Fig. 6. Block diagram of the proposed method.

sensor in each joint. For details on the design and control of the robot, please refer to [2], [1]. Here, we briefly outline the the Cartesian impedance control used for the experiments.

Due to the lightweight design of the LWR-III it is not sufficient to model the robot by a second-order rigid body model. The non negligible joint elasticity between motor and link inertia caused by the Harmonic Drive gears and the joint torque sensor has to be taken into account into the model equation. Following controller structure[3] is realized, which enables high performance impedance control at a rate of 1 kHz.

$$M(\mathbf{q})\ddot{\mathbf{q}} + C(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}) + \boldsymbol{\tau}_{\mathbf{ext}} = \boldsymbol{\tau} \quad (19)$$

$$B_\theta\ddot{\boldsymbol{\theta}} + J(\bar{\mathbf{q}})^T(K_x\tilde{\mathbf{x}}(\bar{\mathbf{q}}) + D_x\dot{\mathbf{x}}(\bar{\mathbf{q}})) + \boldsymbol{\tau} = \bar{\mathbf{g}}(\boldsymbol{\theta}) \quad (20)$$

$$\boldsymbol{\tau} = K(\boldsymbol{\theta} - \mathbf{q}) \quad (21)$$

Following quantities are defining the joint space characteristics of the robot. $\mathbf{q},\boldsymbol{\theta} \in \mathbb{R}^n$ are the link and motor side position. $M(\mathbf{q}) \in \mathbb{R}^{n\times n}$ is the symmetric and positive definite inertia matrix, $C(\mathbf{q},\dot{\mathbf{q}}) \in \mathbb{R}^n$ the Centripetal and coriolis vector, and $\mathbf{g}(\mathbf{q}) \in \mathbb{R}^n$ the gravity vector. $B = \text{diag}\{B_i\} \in \mathbb{R}^{n\times n}$ is the diagonal positive definite motor inertia matrix, which is scaled down with an inner torque control loop to B_θ. $\boldsymbol{\tau} \in \mathbb{R}^n$ is the joint torque and $\boldsymbol{\tau}_{\mathbf{ext}} \in \mathbb{R}^n$ the external torque.

The impedance control is designed with following quantities. $K_x, D_x \in \mathbb{R}^{m\times m}$ are the diagonal positive definite desired stiffness and damping matrix. $\mathbf{x}_d \in \mathbb{R}^m$ is the desired tip position in Cartesian coordinates, and $\mathbf{x}(\bar{\mathbf{q}}) = T(\bar{\mathbf{q}})$ the forward kinematics from joint space to Cartesian coordinates, while $J(\bar{\mathbf{q}}) = \frac{\partial \mathbf{f}(\bar{\mathbf{q}})}{\partial \bar{\mathbf{q}}}$ is the Jacobian of the manipulator. $\bar{\mathbf{q}} = h^{-1}(\boldsymbol{\theta})$ is the static equivalent of $\mathbf{q}$. The gravity compensation term $\bar{\mathbf{g}}(\boldsymbol{\theta})$ is a function of the motor position and is designed in such way, that it provides exact gravity compensation in static case.

B. Static obstacles

The first experiment shows the performance for static obstacles. Billiard balls are arbitrarily arranged on the table and then identified with an object recognition system. Their position is used to define the artificial repulsive potential fields. In Figure 7 (left) the scene view from above is shown, where the robot reached its target configuration. Figure 7 (right) depicts the commanded motion (solid) and the real path of the robot (dashed). v_d was chosen to be 0.2 m/s.

[3]Please not this is a simplified view on the structure, which was chosen for better understanding.

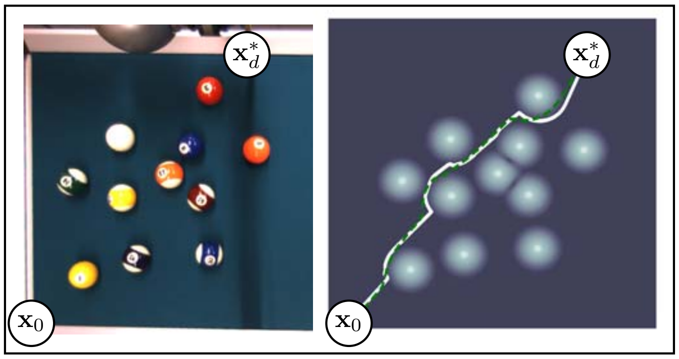


Fig. 7. Configuration of Billiard balls (left). 2D plot of the collision avoidance experiment with the Billiard balls (right).

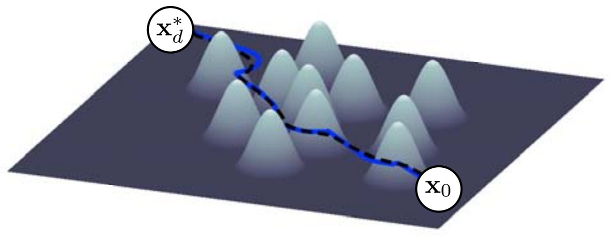


Fig. 8. 3D plot of the collision avoidance experiment with the Billiard balls.

The slight deviation is generated by the use of Cartesian impedance control since no feed forward torque input was used. Figure 8 denotes the 3D visualization and Figure 9 the timely behavior of the robot.

C. Dynamic obstacles

We evaluated the performance of the method for three distinct dynamic situations. In the first one we mounted the DLR 3D Modeler [17] on the robot in order to use the integrated laser scanner for acquiring proximity data. In the second one we used an ART[4] tracking system for passively tracking the human wrist pose. In the third one the estimated external force provided by (7) are chosen as the repulsive input and show how the proposed method can cope with robot-human collisions and unexpected very rigid impacts

[4]**A**dvanced **R**ealtime **T**racking

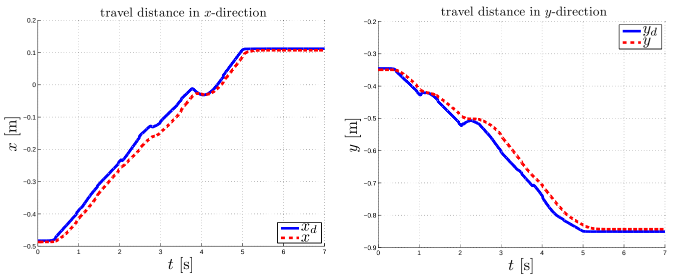


Fig. 9. Time courses of the avoidance in x-direction (left) and y-direction (right). The plot shows the desired trajectory $\mathbf{x}_d$ and the real motion of the robot $\mathbf{x}$.

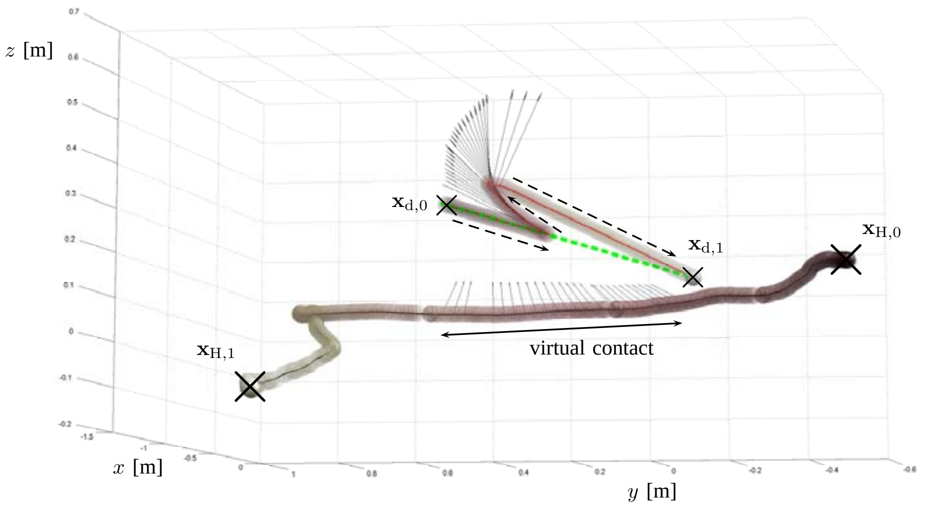


Fig. 10. Dynamic collision avoidance with tracking system.

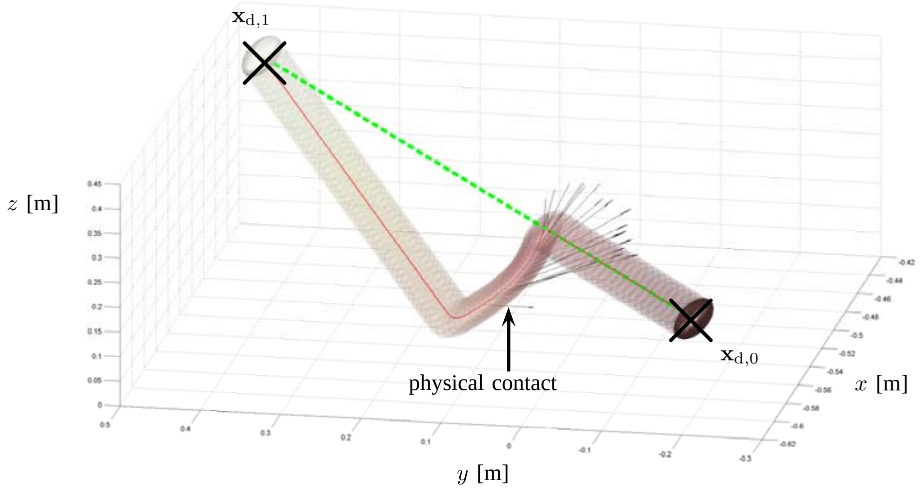


Fig. 11. The plot depicts the behavior for being pushed by a human. After contact is lost, the robot recovers quickly from it and converges to the goal configuration.

with the environment, see Fig 12. The attached video shows all experiments, especially pointing out that the combination of multiple disturbance inputs at the same time can be easily managed.

The result of the second experiment is given in Fig. 10. The robot is commanded to reach the desired goal configuration $\mathbf{x}_d^*$. As soon as the human holds his arm into the workspace and blocks the possible motion path, the robot circumvents the hand and reaches the goal. The original desired motion is depicted (dashed) and the generated virtual forces are shown along the human motion path as well as on the resulting robot trajectory. The human moves from right to left, while the robot intends to reach the right configuration. As soon as the robot is affected by virtual forces it starts deviating from the path and after the human surpassed it, it moves again towards the goal and terminates there.

From Figure 11 one can see how the method can cope with external forces in the same way as with virtual ones. The human pushes the robot while it is moving. The desired motion is deformed such that the robot is deviated from its path, see Figure 11. When contact is lost, the robot converges quickly to the goal again.

In Figure 12 a further response to external contact forces is shown. In the experiment the robot collides with a table after being pushed by the human into this unknown object. Then, the contact information (force magnitude and direction) is used to recover from this second collision. Finally, the robot reaches its goal position.

Especially for the table impact one can see how the Cartesian impedance control, the external force estimation, and the collision avoidance work together to recover from an unexpected very rigid contact, while still reaching the goal.

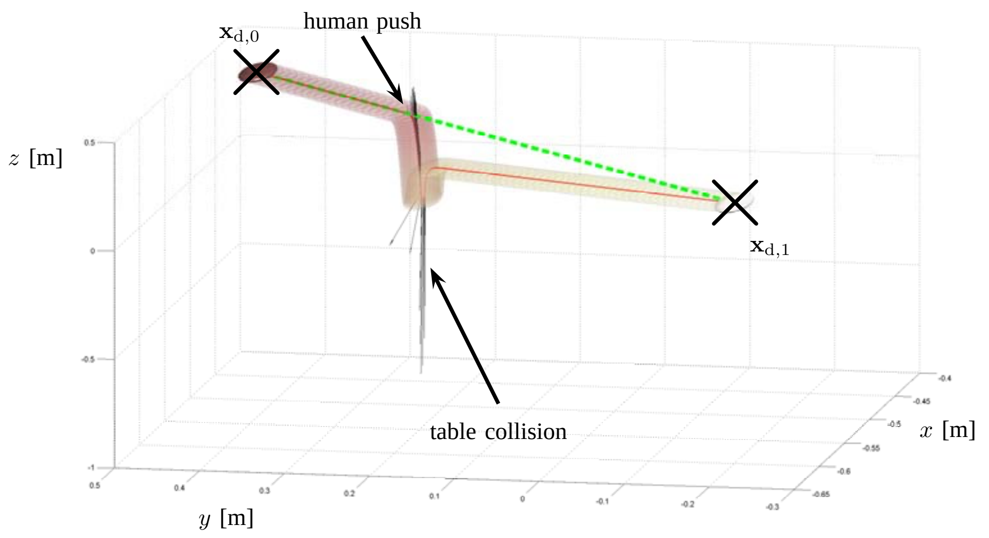


Fig. 12. The plot depicts the behavior for pushing the robot harder, which results in a second collision with the table. Even though the robot has no prior knowledge of the table it quickly recovers from the second contact and finds it way into the final goal.

V. Conclusion

In this paper we outlined a new method for reactive motion generation based on intuitive physical interpretation. The method is well suited to serve in between global motion planning and control to establish well defined and safe behavior even for unexpected virtual and physical contact. It is designed to serve as relief for both sides and provides a smooth motion in complex environments, taking into account both, proximity to objects and external forces. The algorithm allows to command arbitrary velocity profiles to the robot and provides collision avoidance behavior at the same time. Even during circumvention the track speed can be commanded such that no unexpected velocity or acceleration jumps, nor high velocity values may occur.

Acknowledgment

This work has been partially funded by the European Commission's Sixth Framework Programme as part of the project VIACTORS under grant no. 231554. We would like to thank Tim Bodenmüller for his help with implementing the collision avoidance with the DLR 3D Modeler.

References

[1] A. Albu-Schäffer, S. Haddadin, C. Ott, A. Stemmer, T. Wimböck, and G. Hirzinger, "The DLR lightweight robot - lightweight design and soft robotics control concepts for robots in human environments," *Industrial Robot Journal*, vol. 34, no. 5, pp. 376–385, 2007.

[2] A. Albu-Schäffer, C. Ott, and G. Hirzinger, "A Unified Passivity-based Control Framework for Position, Torque and Impedance Control of Flexible Joint Robots," *The Int. J. of Robotics Research*, vol. 26, pp. 23–39, 2007.

[3] O. Brock and O. Khatib, "Elastic Strips: A Framework for Motion Generation in Human Environments," *Int. J. Robotics Research*, vol. 21, no. 12, pp. 1031–1052, 2002.

[4] H. Choset, K. Lynch, S. Hutchinson, G. Kantor, W. Burgard, S. Thrun, and L. Kavraki, *Principles of Robot Motion: Theory, Algroithms, and Implementation*. Cambridge: MIT Press, 2005.

[5] M. V. Damme, B. Vanderborght, B. Verrelst, R. V. Ham, F. Daerden, and D. Lefeber, "Proxy-based sliding mode control of a planar pneumatic manipulator," in *The International Journal of Robotics Research*, 2009, pp. 266–284.

[6] A. De Luca, A. Albu-Schäffer, S. Haddadin, and G. Hirzinger, "Collision Detection and Safe Reaction with the DLR-III Lightweight Manipulator Arm," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS2006), Beijing, China*, 2006, pp. 1623–1630.

[7] S. Haddadin, A. Albu-Schäffer, A. De Luca, and G. Hirzinger, "Collision Detection & Reaction: A Contribution to Safe Physical Human-Robot Interaction," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS2008), Nice, France*, 2008.

[8] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," vol. 2, 1985, pp. 500–505.

[9] J.-C. Latombe, *Robot Motion Planning*. Norwell, MA, USA: Kluwer Academic Publishers, 1991.

[10] H. Minoura, R. Kijima, and T. Ojika, "Collision avoidance using a virtual electric charge in the electrostatic potential field," in *Proceedings of International Conference on Virtual Systems and MultiMedia*, 1996, pp. 289–294.

[11] P. Ögren, N. Egerstedt, and X. Hu, "Reactive mobile manipulation using dynamic trajectory tracking," in *ICRA*, 2000, pp. 3473–3478.

[12] S. Quinlan and O. Khatib, "Elastic bands: connecting path planning and control," in *ICRA*, 1993, pp. 802–807.

[13] J. Roßmann, "Echtzeitfähige kollisionsvermeidende Bahnplanung für Mehrrobotersysteme," Ph.D. dissertation, University of Dortmund, 1993.

[14] ——, "On-Line Collision Avoidance for Multi-Robot Systems: A New Solution Considering the Robots' Dynamics," in *Int. Conf. on Multisensor Fusion and Integration for Intelligent Systems (MFI1996), Washington D.C., USA*, 1996, pp. 249–256.

[15] B. Siciliano and O. Khatib, Eds., *Springer Handbook of Robotics*. Springer, 2008.

[16] L. Singh, H. Stephanou, and J. Wen, "Real-time motion control with circulatory fields," in *ICRA*, 1996, pp. 2737–2742.

[17] M. Suppa, S. Kielhoefer, J. Langwald, F. Hacker, K. H. Strobl, and G. Hirzinger, "The 3D-Modeller: A Multi-Purpose Vision Platform," in *Int. Conf. on Robotics and Automation (ICRA), Rome, Italy*, 2007, pp. 781–787.

[18] C. Warren, "Global path planning using artificial potential fields," in *ICRA*, 1989, pp. 316–321.

[19] Y. Yamamoto and X. Yun, "Coordinated obstacle avoidance of a mobile manipulator," in *ICRA*, 1995, pp. 2255–2260.