Simultaneous Control of Two Robotics Arms Sharing Workspace via Visual Predictive Control

E. Le Flécher, Adrien Durand-Petiteville, Viviane Cadenat, Thierry Sentenac

Simultaneous Control of Two Robotics Arms Sharing Workspace via Visual Predictive Control

Le Flécher E.[1,2], Durand-Petiteville A.[3], Cadenat V.[1,2]and Sentenac T.[1]

[1] CNRS, LAAS, 7 avenue du colonel Roche, F-31400 Toulouse, France
[2] Univ de Toulouse, UPS, F-31400, Toulouse, France
{eleflech,cadenat,sentenac}@laas.fr
[3] Department of Mechanical Engineering, Federal University of Pernambuco UFPE, Recife, Brazil
adrien.durandpetiteville@ufpe.br

Abstract. This work addresses the problem of simultaneously controlling two robotic arms to automatically pick up fruits in an orchard. When considering such a scenario, the design of the controller has to face several challenges: (i) the arms share the same workspace, (ii) the presence of obstacles such as branches and other fruits, and (iii) the system evolves in a dynamic environment, *i.e.,* the positions of the fruits and branches change over time. In this paper, it is proposed to control the two robotic arms relying on a Visual Predictive Control (VPC) scheme. It allows to control the system in a reactive fashion while taking into account a large number of constraints. The global and local models of the VPC scheme as well as constraints dealing with collisions or joint boundaries are presented. Finally, the efficiency of the proposed approach is highlighted with numerous simulation results using the PR2 arms model.

Keywords: Agricultural robotics, multi-arms, Non-linear Model Predictive Control, Image Based Visual Servoing

1 Introduction

To meet the increasing food demands from a growing world population, agriculture will need to double its production by 2050 [11, 14]. This cannot be achieved by simply doubling the inputs (land, water, seeds, labor, etc.) because of constrained resources and environmental concerns. Among the solutions presented in [11, 14] to increase the global agricultural total factor productivity, robotics is one with the potential to effect agriculture in a broad, systemic way, and contribute significantly to meeting our future needs. Up to today, the three most significant impacts robotics and automation have had on agriculture are: (i) Precision agriculture, *i.e.,* the use of sensors to precisely control when and where to apply fertilizers or water; (ii) Auto-guidance on field crop machinery, which today can drive down a field with an accuracy unattainable by human drivers; (iii) Machines that harvest fruits and vegetables for processing (e.g., tomato paste

and orange juice) [3]. Research is now focused on the next wave of sensing, mobility, and manipulation technologies that promise to increase agriculture output and productivity.

Over the last years, one has focused on robotic systems performing weeding, spraying, trimming, or harvesting in orchards. In order to achieve any of these tasks, the following capacities are expected for the robotic system: (i) autonomously navigate in an orchard; (ii) detect and localize a set of natural landmarks of interest; (iii) approach the landmarks with some robotic manipulators; (iv) manipulate/interact with the landmarks of interest without damaging them. After addressing the autonomous navigation problem in [8, 20], the focus is now on the landmark approach problem when considering numerous robotic manipulators embedded on a mobile platform. This problem has already been addressed to design harvesting systems for cucumbers [25] or tomatoes [29]. However, despite the promising results, the obtained harvesting throughput does not match with the farmers needs and must be increased [28]. To achieve this aim, one of the most adopted approach consists of embedding several robotic arms on the mobile platform, as it is done in [28] for example. However, the throughput increase is not proportional to the number of manipulators, it also has to do with their distribution on the machine and their workspace. Indeed, it is pointed in [26] that fruits are not uniformly distributed on commercial orchards trees. To maximize the fruits accessibility, growers use specific trees with high fruits density, thus facilitating the manual harvesting. As a consequence, when designing a harvesting embedding multiple arms, it is also important to consider the workspace of each arm and the possible overlapping. Thus, even if the arms do not collaborate to manipulate one single object, they share a common workspace. It is then possible to use several arms to pick fruits in the same high density area, which increases the overall throughput.

In this work, it is proposed to design a control scheme for two robotic arms aiming at positioning their end effector to achieve an agricultural task in an orchard. When considering performing a task in orchard, it seems relevant to focus on reactive controllers. Indeed, a tree is a highly dynamic environment because of the flexibility of the branches and the weight of the fruits: each time a fruit is harvested, the local distribution of fruits and branches is modified. Among the available control scheme, it is proposed to use an image based visual servoing (IBVS) scheme [6]. It requires the use of a camera, and nowadays it is simple to obtain a camera small enough to be install close to the end-effector without creating any trouble. Moreover, it seems to be the appropriate sensor to track the fruit of interest. IBVS consists of computing the error between the current coordinates of points of interest and a set of desired values. This error is then minimized via a controller based on the image Jacobian matrix, allowing to control the displacements of the end effector. Computing the control law directly in the image space makes this approach robust to modeling errors and reactive objective modification, however it does not provide any guarantee regarding the trajectory of the camera in the workspace. This represents a major issue when considering arms sharing a common workspace.

Numerous works have been developed in order to modify the camera trajectory in the workspace or to handle issues such as the loss of the visual features, collisions with obstacles,... For example, the work presented in [12] proposes to use advanced visual features to obtain more appropriate camera trajectories. However, they do not guarantee the non-collision with obstacles. Another approach consists of coupling IBVS with other elementary controllers, and to switch from one controller to the other based on the current situation (obstacles, occultations, joint boundaries, ...) [4] [9] [7]. This approach requires to design numerous *ad-hoc* controllers, and defining the switching rules can happen to be tenuous. Other works use the robot redundancy to modify the trajectory in order to satisfy the given constraints [22] [5]. However, this method can only be used if the main task does not use all the degrees of freedom [21], which might be the case when using systems with large number of degrees of freedom, such as humanoid robots. Finally, it has been proposed to couple IBVS with path planning [24], [18], in order to generate a path free of occultations and collisions. However, to do so, it is mandatory to have a model of the environment prior to the task execution, which might be challenging in a dynamic environment.

In this work, it is proposed to rely on a well adopted solution allowing to take into account constraints on position, velocity, or workspace: the model predictive control scheme [15]. This approach relies on the minimization of a cost function made of the predicted states of the system and defining the task to achieve. Moreover, when minimizing the cost function with a numerical solver, this latter can handle numerous constraints [15]. Thus, to obtain a reactive controller dealing with constraints, it is proposed to design an IBVS-based NMPC (Nonlinear Model Predictive Control) controller, also named Visual Predictive Control (VPC). For this particular case, the cost function is defined as the error between the image coordinates of the visual features representing the landmark of interest at the current pose and the desired one. To develop a VPC scheme, it is then mandatory to (i) obtain the prediction models of the system state, (ii) design the cost function defining the task to achieve, (iii) design the mandatory constraints to achieve the task, and (iv) select the numerical solver allowing to minimize the cost function under the given constraints. Thus, this paper addresses the aforementioned points for the specific case of two simultaneously controlled robotic arms. Moreover, it is an extended version of the work presented in [10] which has been complemented with an analysis of the related works, a discussion regarding the terminal constraint and a more comprehensive set of simulation results.

The paper is organized as follows. First, section II is dedicated to an overview of the existing VPC scheme. Next, in section III, one presents the different used predictive models, before addressing in section IV the design of the cost function and of the different constraints. In section V, a set of simulations is presented to support the proposed approach and highlight its efficiency. Finally, in section VI, one summarizes and analyses the obtained results before developing the expectations for the future.

2 Related Works

As it has been explained, NMPC schemes require to use a model to predict the future states of the system. In the case of VPC, it consists of predicting the coordinates of the visual features knowing a sequence of control inputs. Several works have addressed the design of VPC schemes, and according to [1], they can be classified into two categories: the local and global models. The first ones relies on the interaction matrix (or image Jacobian) [6], whereas the second ones use the camera pose. For both cases, the models are built considering a flying camera, *i.e.,* the system embedding the camera is not taken into account in the model. Finally, none of the models proposes an analytic form for the prediction of the visual features. Thus, the predicted values are obtained using a linear approximation, or with advanced numerical integration schemes, such as Runge-Kutta.

Most of the existing works belong to the first category, *i.e.,* relying on local models. For example, in [2] the authors use a first order approximation to control a camera embedded on a robotic arm. Similar approaches are presented in [17] for a flying camera and in [19] for a mobile robot. In [27] and [16], the visual servoing scheme is represented by a linear parameter varying (LPV) system. The objectives of these works is to demonstrate the feasibility of VPC subject to constraints on the camera field of view or command vector. The results show the relevance of the approach, but are obtained for small displacements between the initial and desired pose, minimizing the impact of the predictions. On the opposite, the work presented in [23] proposes to control an UAV while avoiding other aircraft. In this case, the distance between the vehicles is large enough to use an approximate value of the depth in place of the real one.

In this work, it is proposed to first perform an evaluation of the accuracy of the prediction models using the real or approximated depth values. This evaluation is done in the context of a camera embedded on a robotic arm. Following this evaluation, we then present the design of our VPC scheme to simultaneously control two robotic arms.

3 Modeling

In this section, one first focuses on the modeling of the system, *i.e.,* the robotic arms, the cameras and the targets [10]. Then, two models to predict the visual feature coordinates of a point are presented [1] [10]. One ends with a discussion regarding the viability of these models for the visual predictive control problem.

3.1 System modeling

The robotic system considered in this work is composed of two identical arms, each of them containing m_q revolute joints. A camera is attached to the end effector of each arm. To model the system, let us first define a global frame $F_b = (O_b, \mathbf{x}_b, \mathbf{y}_b, \mathbf{z}_b)$ attached to the system base. Moreover, two frames $F_{c_i} =$

$(O_{c_i}, \mathbf{x}_{c_i}, \mathbf{y}_{c_i}, \mathbf{z}_{c_i})$, with $i \in [1,2]$, are used to represent the pose of the camera attached to the i^{th} arm.

Let define q_{ij} and $\dot{q}_{ij}$, with $j \in [1, ..., m_q]$, respectively the angular position and velocity of the j^{th} joint of the i^{th} arm. Thus, for the i^{th} arm, one obtains the configuration vector $Q_i = [q_{i1}, q_{i2}, ..., q_{im_q}]^T$ and the control input vector $\dot{Q}_i = [\dot{q}_{i1}, \dot{q}_{i2}, ..., \dot{q}_{im_q}]^T$. Finally, for the whole system one obtains $Q = [Q_1^T, Q_2^T]^T$ and $\dot{Q} = [\dot{Q}_1^T, \dot{Q}_2^T]^T$.

The cameras mounted on the end effectors are modeled using the pinhole model. Thus, the projection matrices H_{I_i/c_i} mapping the 3D coordinates (x, y, z) of a point in the camera frame F_{c_i} to its 2D projection (X, Y) on the image plan F_{I_i} is defined as follows:

$$\begin{bmatrix} X \\ Y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} f/z & 0 & 0 & 0 \\ 0 & f/z & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{1}$$

In this work, the arms are controlled to make each camera reach a pose defined by the coordinates of point visual features in the image space. To do so, one uses two landmarks made of four points. When considering the i^{th} camera/target couple, the coordinates of the projection of each point on the image is denoted $S_{il} = [X_{il}, Y_{il}]^T$, with $i \in [1,2]$ and $l \in [1,2,3,4]$. Thus, the visual features vector for each camera is defined as $S_i = [S_{i1}, S_{i2}, S_{i3}, S_{i4}]^T$, whereas the system one is given by $S = [S_1^T, ..., S_{n_a}^T]^T$. In the same way, a vector for the visual features desired coordinates of each camera is defined as $S_i^* = [S_{i1}^*, S_{i2}^*, S_{i3}^*, S_{i4}^*]^T$.

3.2 Prediction of the visual features

A visual predictive control scheme requires to be able to predict the values of the visual features for a given sequence of control inputs. In [1], the authors present two ways to perform the prediction step for a flying camera: the local and global approaches. This section is devoted to the presentation of both methods for the case of a 6 degrees of freedom camera embedded on a robotic arm.

The global model Global models rely on the pose of the camera. It can be decomposed as follows: from the image coordinates, the location of the visual features in the current camera frame is calculated (2D to 3D); the visual features coordinates are then computed in the predicted camera frame (3D to 3D); and finally the visual features are projected on the predicted image plane (3D to 2D). To do so, the camera pose is generally predicted by integrating the camera kinematic screw [1]. In this work, one integrates the robotic arm, which allows to predict the camera pose based one the robot command vector, minimizing the approximation. To do so, let us consider a discrete time system with a sampling period T_s, where $t_{k+1} = t_k + T_s$, and the notation $.(k) = .(t_k)$. The configuration $Q_i(k+1)$ of the i^{th} arm, with $i \in [1,2]$, at the next iteration obtained after

applying the control inputs $\dot{Q}_i(k)$ is given by:

$$Q_i(k+1) = Q_i(k) + \dot{Q}_i(k)T_s \tag{2}$$

Knowing the arm configuration at instants t_k and t_{k+1}, it is then possible to compute the homogeneous transformation matrices between the camera frame F_{c_i} and the base one F_b.

$$\bar{o}_{F_b}(.) = H_{b/c_i}(.)\bar{o}_{F_{c_i}}(.) \tag{3}$$

where $\bar{o}$ are the homogeneous coordinates of a point feature expressed in the base or camera frame. Finally, coupling (3) with (1), one obtains a prediction of the coordinates of a point feature in the image space.

$$\begin{aligned} &S_{il}(k+1) = \\ &H_{I_i/c_i}(k+1)H^{-1}_{b/c_i}(k+1)H_{b/c_i}(k)H^{-1}_{I_i/c_i}(k)S_{il}(k) \end{aligned} \tag{4}$$

The local model Local models rely on the differential equation mapping the derivative of the visual feature coordinates $\dot{S}_{il}$ to the robot control input vector $\dot{Q}_i$. To obtain this equation, one uses the robot's Jacobian J_i, mapping the camera kinematic screw T_{c_i} and $\dot{Q}_i$ (5), and the interaction matrix L_{il}, mapping $\dot{S}_{il}$ to T_{c_i} (6).

$$T_{c_i} = J_i\dot{Q}_i \tag{5}$$

$$\dot{S}_{il} = L_{il}T_{c_i} \tag{6}$$

where

$$L_{il} = \begin{bmatrix} -\frac{1}{z_l} & 0 & \frac{X_l}{z_l} & X_lY_l & -(1+X_l^2) & Y_l \\ 0 & -\frac{1}{z_l} & \frac{Y_l}{z_l} & 1+Y_l^2 & -X_lY_l & X_l \end{bmatrix} \tag{7}$$

By combining equations (5) and (6), one obtains the following local model for each arm/camera couple:

$$\dot{S}_{il} = L_{il}\ J_i\ \dot{Q}_i \tag{8}$$

Usability of the models When considering using one of the models on an experimental system, several aspects have to be taken into account. First, both models require a measure/estimate of the visual feature depth z (see equations (1) and (7)). However, when using a monocular camera, this value can only be retrieved by estimation. Another widely adopted option consists of using a constant value, such as the depth value at the initial or desired pose [6]. The second issue is related to the integration of the local model to obtain the prediction equation. A first option consists of using an advanced numeric scheme to accurately compute the integration, whereas a second one relies on a linear approximation of the model, such as:

$$S_{il}(k+1) = S_{il}(k) + L_{il}J_i\dot{Q}_i(k)T_s \tag{9}$$

Thus, depending on the user choices regarding these issues, the prediction feature coordinates will be more or less accurate. In section 5, different configurations will be evaluated for our specific application in order to select the most appropriate and realistic solution.

4 Controller Design

In this section the control of the arms motion to reach their own targets and compounding necessary constraints are presented. To achieve these goals, NMPC control has been implemented to find the optimal inputs vector which minimizes a cost function subject to constraints. To do so, we first present the cost function to be minimized in order to reach the desired targets. Then, the expression of each constraint will be defined. A similar approach was presented in [10] for a multi-arm robotic system.

4.1 Visual Predictive Control

The proposed VPC scheme couples NMPC (Nonlinear Model Predictive Control) with IBVS (Image-Based Visual Servoing). Thus, similarly to NMPC, it consists of computing an optimal control sequence $\bar{\dot{Q}}(.)$ that minimizes a cost function J_{N_P} over a prediction horizon of N_P steps while taking into account a set of user-defined constraints $C(\bar{\dot{Q}}(.))$. Moreover, similarly to IBVS, the task to achieve is defined as an error in the image space. Thus, the cost function to minimize is the sum of the quadratic error between the visual feature coordinates $\hat{S}(.)$ predicted over the horizon N_p and the desired ones S^*. It then possible to write the VPC problem as:

$$\bar{\dot{Q}}(.) = \min_{\dot{Q}(.)} \Big(J_{N_P}(S(k), \dot{Q}(.)) \Big) \tag{10}$$

with

$$J_{N_P}(S(k), \dot{Q}(.)) = \sum_{p=k+1}^{k+N_p-1} [\hat{S}(p) - S^*]^T [\hat{S}(p) - S^*] \tag{11}$$

subject to

$$\hat{S}(.) = (4) \text{ or } (8) \tag{12a}$$

$$\hat{S}(k) = S(k) \tag{12b}$$

$$C(\bar{\dot{Q}}(.)) \leq 0 \tag{12c}$$

As mentioned in the previous section, the visual feature coordinates can be predicted either using the global or local models (equation (12a)). To compute the prediction, one uses the last measured visual features (equation (12b)). Finally,

equation (12c) defines the set of constraints to be taken into account when minimizing the cost function (11).

Thus, solving equation (10) leads to the optimal sequence of control inputs $\bar{\dot{Q}}(.)$. In this work, and as it is usually done, only the first element $\bar{\dot{Q}}(1)$ of the sequence is applied to the system. At the next iteration, the minimization problem is restarted, and a new sequence of optimal control inputs is computed. This loop is repeated until the task is achieved, *e.i.,* the cost function (11) is vanished.

4.2 Constraints

As seen in the previous section, it is possible to add a set of constraints in the minimization problem. In this section, one presents the constraints related to the geometry of the robot, the velocity of the joints, and the field of view of the camera. One finishes with defining a set of constraints that guarantee the non-collision of the arms despite sharing a common workspace.

System constraints The joint velocity lower and upper bounds are respectively denoted $\dot{Q}_{min}$ and $\dot{Q}_{max}$. They are two $N_{\dot{Q}}$ long vectors, with $N_{\dot{Q}} = 2m_qN_p$. They allow to define the lower and upper limits of each joint of each arm over the prediction horizon. Consequently, the constraint on $\dot{Q}(.)$ is described in the domain $\kappa = [\dot{Q}_{min}, \dot{Q}_{max}]$, and the velocity constraint is then written as:

$$\begin{bmatrix} \dot{Q} - \dot{Q}_{max} \\ \dot{Q}_{min} - \dot{Q} \end{bmatrix} \leq 0 \tag{13}$$

Mechanical limits follow the same principle. The lower and upper bounds of the angular values are denoted Q_{min} and Q_{max}. They are two N_Q long vectors, with $N_Q = 2m_qN_p$. These constraints are then given by:

$$\begin{bmatrix} Q - Q_{max} \\ Q_{min} - Q \end{bmatrix} \leq 0 \tag{14}$$

Finally, the visual constraints are defined by the image dimension. One denotes the bounds by S_{min} and S_{max}. They are two N_S long vectors, with $N_S = 8n_aN_p$. Thus, all the predicted visual features coordinates need to be kept into this space, and the constraints can then be written as:

$$\begin{bmatrix} S - S_{max} \\ S_{min} - S \end{bmatrix} \leq 0 \tag{15}$$

Overlapping workspace constraint Due to the overlapping workspace, it is mandatory to guarantee the non-collision between the arms. To do so, one computes the shortest distance $d_{i,j|i',j'}$ between the j^{th} joint of the i^{th} arm and the j'^{th} joint of the i'^{th} arm. The closest distance is evaluated as the norm of

the segment perpendicular to both assessed links as it is illustrated in figure 1. Finally, in order to avoid collisions, one defines the following constraint:

$$\left[d_{min} - d\right] \leq 0 \tag{16}$$

where $d = [d_{1,1|2,1}, ..., d_{1,m_q|2,m_q}]$ and d_{min} is a pre-defined safe distance.

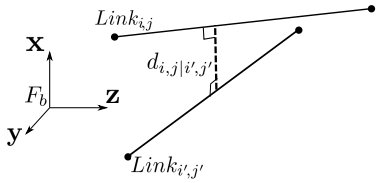


Fig. 1. Computation of the shortest distance $d_{i,j|i',j'}$ between links L_{ij} and $L_{i'j'}$. Source [10]

Terminal constraint When using a NMPC scheme, it is necessary to guarantee that the set of control inputs makes the camera converge towards the desired pose. Due to the fact that a NMPC scheme minimizes the distance between a predicted trajectory and a desired one, there is no guarantee that the ultimate predicted state has converged towards the desired one. Moreover, the prediction horizon might be too short or the constraints on the control inputs might be too restrictive to reach the goal. [15]. Thus, to check the feasibility, one adds a final constraint in the optimization problem. It is defined as the error between the prediction of the visual feature coordinates $\hat{S}(k + N_p - 1)$ obtained at the end of the prediction horizon, and the desired ones S^*. If the solver cannot compute a sequence of control inputs $\bar{\dot{Q}}$ that respects this constraint, then the problem is not feasible, and there is no guarantee regarding the system convergence.

$$|\hat{S}(k + N_p - 1) - S^*| - \delta_{tc} \leq 0 \tag{17}$$

where δ_{tc} is a user defined threshold. Two remarks can be made concerning the use of the terminal constraint. First, none of the works studied during the literature review proposes to include a terminal constraint, despite its ease of implementation and utility. Moreover, authors usually weight the last predicted value based on the distance to the desired one. This method helps the solver to converge towards an optimal solution but it does not guarantee the convergence as the terminal constraint does. Second, the terminal constraint guarantees the convergence of the predicted states towards the desired ones. In the case the predictions are strongly erroneous, then the real system cannot converge towards the desired sate. Thus, using a terminal constraint does not provide an absolute guarantee of the task realization.

All the constraints are now defined and can be integrated into the non-linear function $C(\bar{\dot{Q}}(.))$:

$$C(\bar{\dot{Q}}) \leq 0 \Rightarrow \begin{bmatrix} \dot{Q} - \dot{Q}_{max} \\ \dot{Q}_{min} - \dot{Q} \\ Q - Q_{max} \\ Q_{min} - Q \\ S - S_{max} \\ S_{min} - S \\ \hat{D}_{min} - D \\ |\hat{S}(k + N_p - 1) - S^*| - \delta_{tc} \end{bmatrix} \leq 0 \tag{18}$$

Thus, the optimal input must belong to the domain defined by $C(\bar{\dot{Q}}(.))$.

5 Simulation

In this section, one presents the results obtained by simulating the control of two arms with MATLAB software. A 7 DOF model has been selected, based on the arms embedded on the PR2 robot. In these simulations, the task consists of positioning each camera with respect to a given target. Each final pose is defined in the image space by $S_i^* = [x_1*, y_1*, x_2*, y_2*, x_3*, y_3*, x_4*, y_4*]^T$, which are the coordinates of four points representing the corresponding target. Moreover, one setups the prediction horizon to $N_p = 5$. Regarding the constraints, the angular velocity limits for the first prediction step are given by $\kappa = [-0.2\ 0.2]$ for each joint, whereas it is extended to $\kappa = [-5\ 5]$ for the rest of the prediction step. This allows to decrease the computation time while guaranteeing a feasible control input for the first step. The size of the image is $S_{min} = [-4\ -4]$ and $S_{max} = [4\ 4]$. The angular constraints on each joint are setup as described in the PR2 documentation [13]. Moreover, the minimum collision avoidance distance is fixed to $D_{min} = 0.1$. The initial coordinates of the visual features are computed based on the initial configuration of the robot given by $q_{init} = [0\ \frac{\pi}{2}\ 0\ 0\ 0\ 0\ 0\ 0\ \frac{\pi}{2}\ 0\ 0\ 0\ 0\ 0]^T$ for every simulation. Finally, one fixes $T_s = 50ms$.

5.1 Comparison between the different prediction models

The first set of simulation assesses the accuracy of different predictive models presented in section 3.2. To do so, the two robotics arms are controlled via VPC from q_{init} to $q^* = [-\frac{\pi}{8}\ \frac{\pi}{2}\ 0\ 0\ 0\ 0\ 0\ 0\ \frac{\pi}{2}\ 0\ 0\ 0\ 0\ 0]^T$, using the global model and the real depth values. When simulating a VPC scheme, this configuration can be considered as the ground truth, as there is no approximation in the calculations. Next, at each iteration, the optimal sequence of control inputs calculated by the VPC scheme, is used to compare the different prediction models. To do so, the predicted values at the end of the prediction horizon using a particular model are compared with the reference ones.

In figure 2, one can see the evolution of the quadratic error between the predicted coordinates of one visual feature and the ground truth. Six models are presented by combining the global model (solid lines), local Euler approximation (dashed lines) and local Runge Kutta approximation (dotted lines) with the depth values real z (black) and desired z^* (blue). First, it can be seen that the error decreases over the servoing. Indeed, when the camera is close to the desired pose, the sequence of optimal control inputs is quite small, as there is no much distance left to perform. Having small control inputs then reduces the error in the predicted values. Moreover, at the end of the simulation, $z(t)$ and z^* are very similar, which explains the convergence of the blue and black curves for each model. Next, it can be noticed that the global models and the local Runge Kutta are significantly more accurate than the local Euler one. Indeed, using this latter prediction scheme introduces a large error, and might not allow the system to converge toward the desired configuration. In figure 3, the errors are summed over the simulation to offer a better view of model performances. When considering the use of the real depth, the global model is the most accurate. However, this result is obtained in a simulation, and its use in a real context would require an estimator of the depth. Thus, one now considers one more realistic scenario where the depth is approximated with the constant value z^*. In this case, the local Runge Kutta is the most accurate model. Thus, for the rest of the simulations the local Runge Kutta model with z^* is used to evaluate the performances.

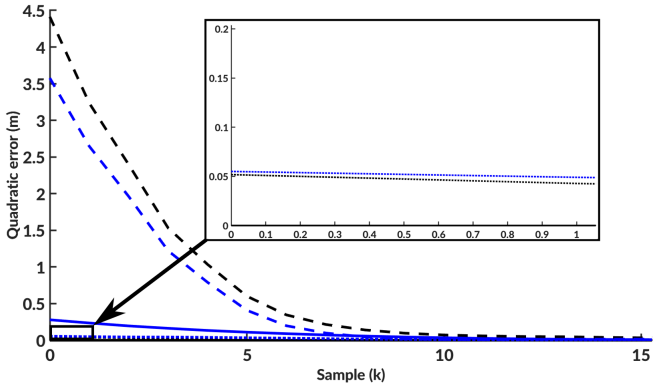


Fig. 2. Quadratic errors of prediction over the simulation - Solid: Global model - Dashed: Local Euler - Dotted: Local Runge Kutta - Black: z - Blue: z^*. Source [10]

5.2 Collision-free scenario

In this second simulation, the system has to reach the desired configuration $q^* = [-\frac{\pi}{16}\ \frac{3\pi}{8}\ 0\ \frac{\pi}{4}\ 0\ \frac{\pi}{8}\ 0\ \frac{\pi}{16}\ \frac{\pi}{2}\ \frac{\pi}{4}\ \frac{\pi}{8}\ 0\ 0\ 0]^T$ from the initial one q_{init} while

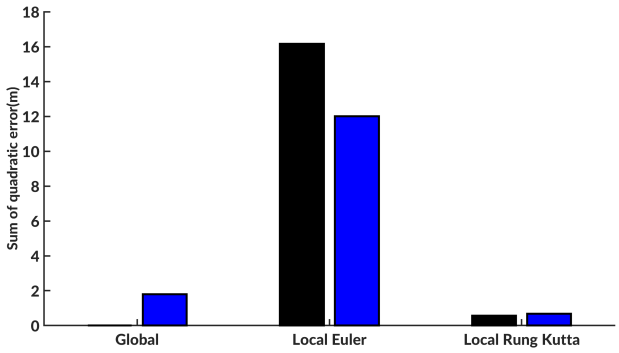
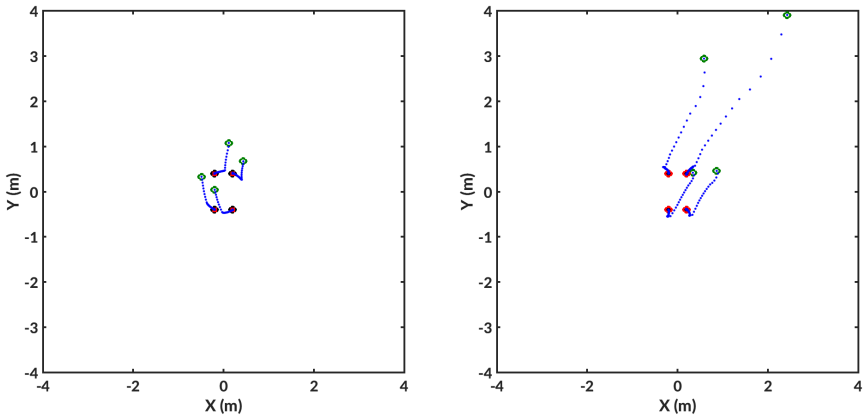


Fig. 3. Sum of the quadratic errors of prediction for global, local Euler and local Runge-Kutta methods - Black: z - Blue: z^*. Source [10]

dealing with the previously presented constraints. The VPC scheme relies on the local model coupled with a Runge Kutta algorithm and the desired depth z^*. Using this configuration, the VPC scheme achieves to control the robotic system to converge towards the desired poses, as it can be seen in the different results presented hereafter. First, the evolution of visual features is displayed in figure 4 and highlights the achievement of the task for both cameras. Indeed, the proposed scheme was able to control both arms to make the current visual features converge towards the desired ones from their initial locations (respectively represented by the red and green crosses in figures 4(a) and 4(b)). To achieve the task, the arms had to reach the configurations presented in figure 5. Regarding the constraints, the image limits were never reached and there was no risk of collision during the servoing. Moreover, it can be seen in figures 6(b) and 6(a) that the joint angular values remain within the defined domain. Similarly, in figures 6(d) and 6(c), one can see that the the joint velocities were bounded by the constraints. Indeed, the command inputs stay within the given range $\kappa = [-0.2\ 0.2]$ all along the simulation.

5.3 Auto-collision scenario

This new simulation is setup in order to create a collision between the two robotic arms. Thus, the system has to reach the desired configuration $q^* = [\frac{\pi}{16}\ \frac{3\pi}{8}\ 0\ \frac{\pi}{4}\ 0\ \frac{\pi}{8}\ 0\ -\frac{\pi}{16}\ \frac{\pi}{2}\ \frac{\pi}{4}\ \frac{\pi}{8}\ 0\ 0\ 0]^T$. Moreover, as previously done, one uses the Runge Kutta local model coupled with the desired depth values. As it can be seen in figure 7, one more time the desired visual features are reached from the initial poses. However, as shown in figure 8, the arms had to cross each other to reach the desired locations. Thanks to the shared environment constraint included in the control problem, the collision was avoided. Indeed, figure 9 shows that the distance between the different links was never smaller than the given value $D_{min} = 0.1m$. In addition to the auto-collision avoidance,



(a) Left camera (b) Right camera

Fig. 4. Evolution of the visual features using local Runge-Kutta model and z - Blue dotted: Trajectories - Red circles: Final locations - Green circles: Initial locations. Source [10]

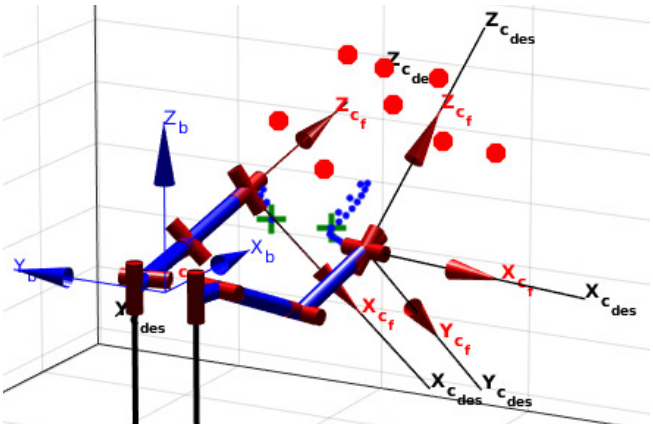
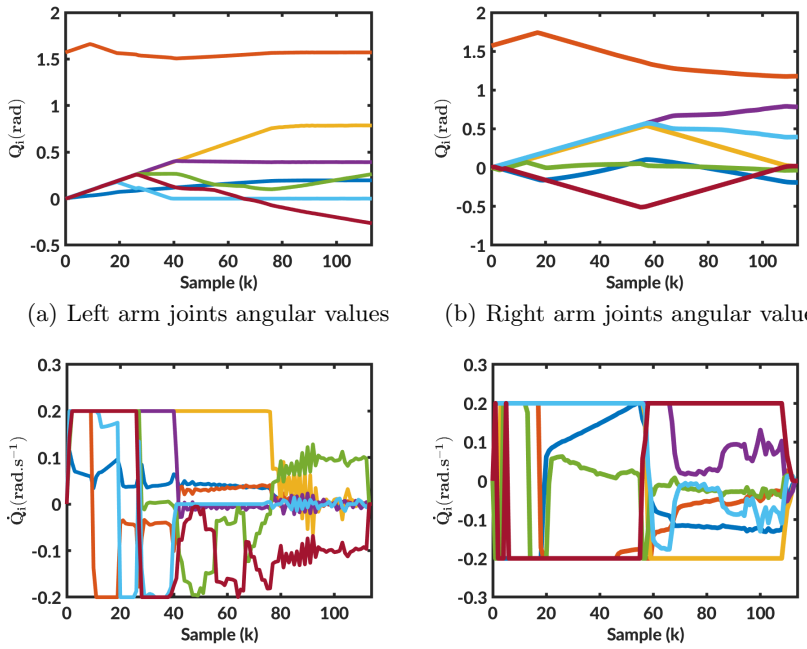


Fig. 5. Final poses of the arms - Red dots represent the two targets - Green crosses are the initial poses of the end effectors. Source [10]

the other constraints on the joints angular values and velocities were respected (see figures 10 and 11).

5.4 External collision scenario

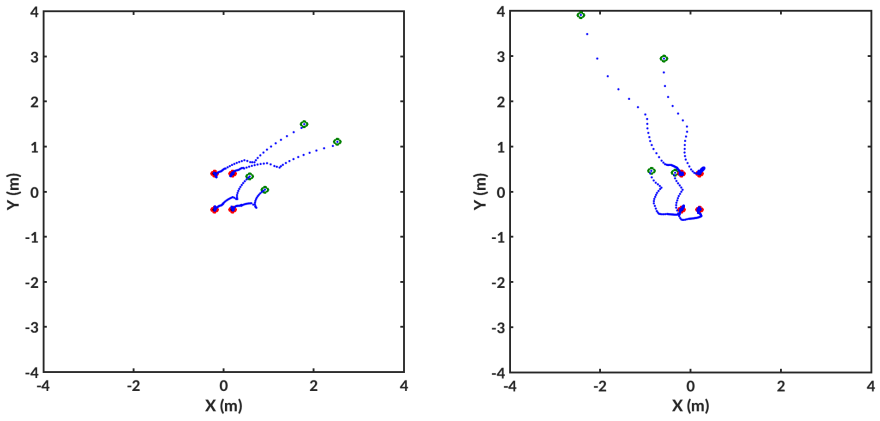
In this scenario, the goal is to achieve the task while dealing with an obstacle on the right arm trajectory. First, one defines the desired pose such as $q^* = [-\frac{\pi}{16}\ \frac{3\pi}{8}\ 0\ \frac{\pi}{4}\ 0\ \frac{\pi}{8}\ 0\ \frac{\pi}{16}\ \frac{\pi}{2}\ \frac{\pi}{4}\ \frac{\pi}{8}\ 0\ 0\ 0]^T$. Next, an obstacle is positioned in the



(a) Left arm joints angular values (b) Right arm joints angular values

(c) Left arm joints angular velocities (d) Right arm joints angular velocities

Fig. 6. Joint configurations and velocities. Source [10]



(a) Left camera (b) Right camera

Fig. 7. Evolution of the visual features using local Runge-Kutta model and z - Blue dotted: Trajectories - Red circles: Final locations - Green circles: Initial locations. Source [10]

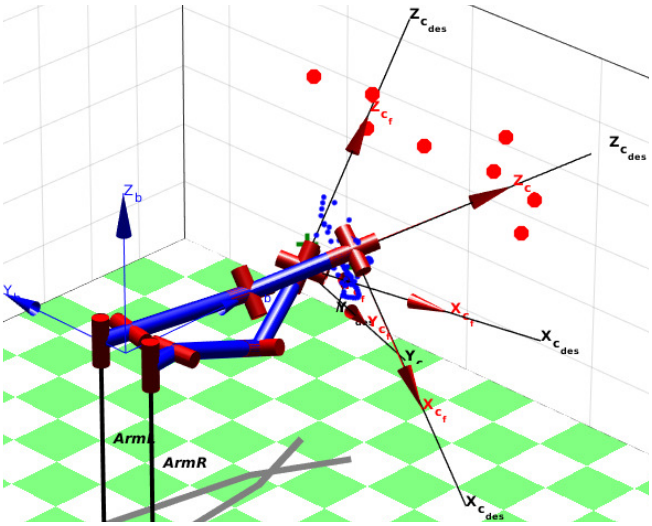


Fig. 8. Final poses of the arms - Red dots represent the two targets - Green crosses are the initial poses of the end effectors. Source [10]

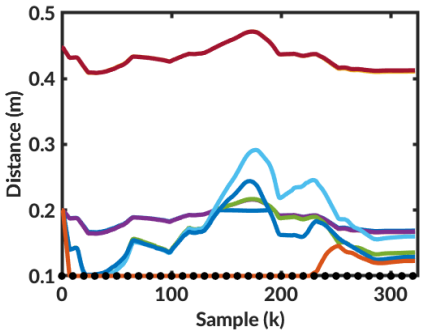
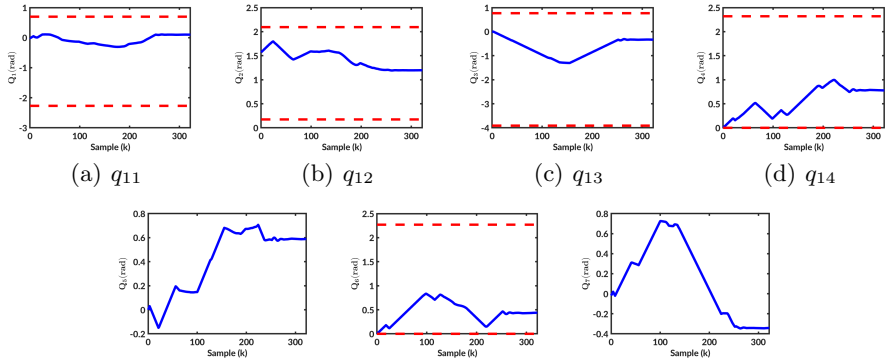


Fig. 9. Collision distance - Solid: Distances between links - Dotted: Minimal allowed distance. Source [10]

scene at the coordinates $\bar{O}_{Obs|_{F_b}} = (0.79, -0.27, -0.08)$ (see figure 12). Finally, the minimal distance is setup to 0.1m.

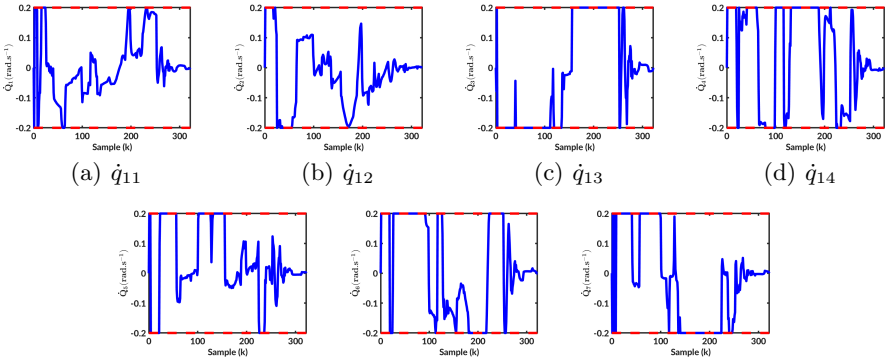
As it can be seen in figure 13(a), the positioning task is achieved, and the visual features have reached the desired coordinates. Moreover, the corresponding trajectory is plotted in figure 12. It should be noticed that the obstacle on the right arm trajectory has been avoided thanks to the distance constraint. Indeed, as it can be seen in figure 13(b), the distance between any segment of the arm and the obstacle is always higher than the defined threshold. Thus, it is possible to safely achieve the positioning task despite the presence of unknown obstacle.



(a) q_{11} (b) q_{12} (c) q_{13} (d) q_{14}

(e) q_{15} (f) q_{16} (g) q_{17}

Fig. 10. Configuration of the right arm over the simulation - Blue solid: Angular joint value q_{1j} - Red dashed: Angle boundaries. Source [10]



(a) $\dot{q}_{11}$ (b) $\dot{q}_{12}$ (c) $\dot{q}_{13}$ (d) $\dot{q}_{14}$

(e) $\dot{q}_{15}$ (f) $\dot{q}_{16}$ (g) $\dot{q}_{17}$

Fig. 11. Right arm joints velocities - Blue solid: Joint velocity value - Red dashed: Velocity boundaries. Source [10]

5.5 Auto/External-collision scenario

This simulation aims at showing the ability of the system to avoid both a collision between the arms and with an external obstacle. To do so, one setups $q_{init} = [-\frac{\pi}{8}\ \frac{3\pi}{8}\ 0\ \frac{\pi}{4}\ 0\ \frac{\pi}{8}\ 0\ \frac{\pi}{16}\ \frac{\pi}{2}\ \frac{\pi}{4}\ \frac{\pi}{8}\ 0\ 0\ 0]^T$. Moreover, the desired configuration is given by $q^* = [\frac{\pi}{20}\ \frac{\pi}{2}\ -\frac{\pi}{2}\ \frac{\pi}{6}\ \pi\ \frac{\pi}{4}\ 0\ -\frac{\pi}{20}\ \frac{\pi}{2}\ \frac{\pi}{2}\ \frac{\pi}{6}\ 0\ 0\ 0]$. The obstacle is positioned at the coordinates $\bar{O}_{Obs\ F_b} = (0.79, -0.27, -0.08)$ (see figure 14) and the minimal distance for both arms and obstacle is set up to 0.1m.

One more time, the task is successfully achieved as it can be seen in figure 15. Indeed, the visual features manage to converge towards the desired values despite the presence of an obstacle and the possible auto-collision. The corresponding trajectory is plotted in figure 14. This trajectory is obtained by taking into

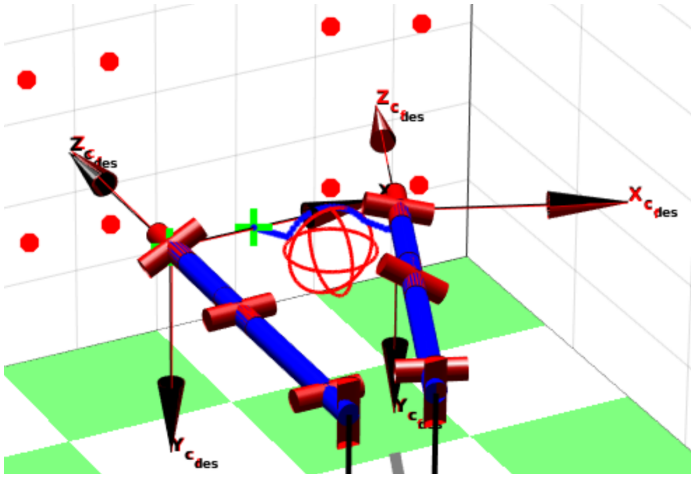
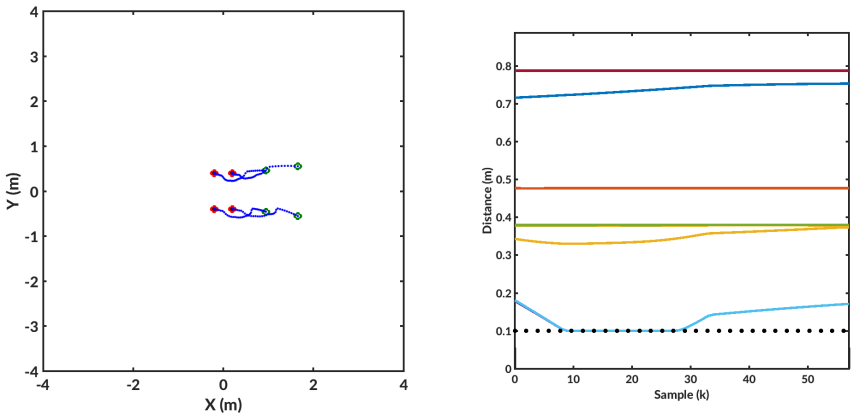


Fig. 12. Final configurations of the robotic arms



(a) Evolution of the visual features in the right camera - Blue dotted: Trajectories - Red circles: Final locations - Green circles: Initial locations

(b) Distance between the obstacle and the different links of the right arm - Solid: Distances - Dotted: Minimal allowed distance

Fig. 13. Evolution of the visual features and distance between the right arm links and the obstacle

account both the constraints related to the arms distances and to the obstacle distance (see figures 16(a) and 16(b)).

6 Conclusion

In this paper, the problem of fruit picking using a dual-arm robot in a commercial orchard has been considered. A strategy based on VPC has been developed

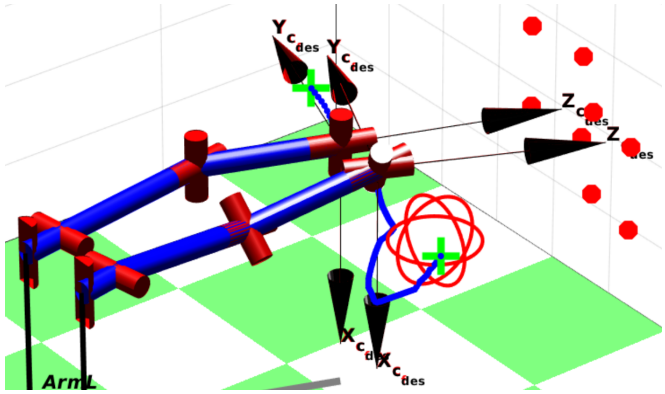
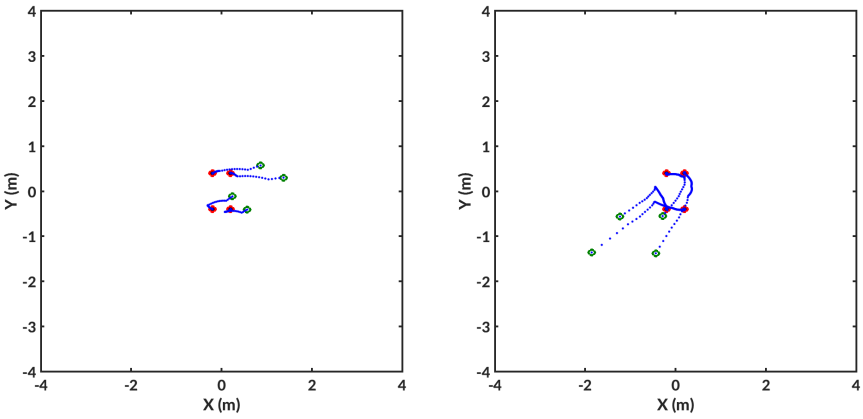


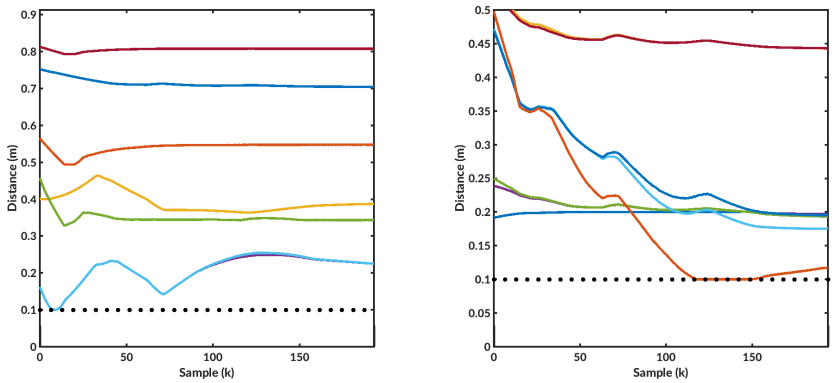
Fig. 14. Final configurations of the robotic arms



(a) Left camera (b) Right camera

Fig. 15. Evolution of the visual features in the right camera - Blue dotted: Trajectories - Red circles: Final locations - Green circles: Initial locations

to make its end-effectors reach a desired pose close to the corresponding fruits, while respecting constraints due to the environment dynamics, the visual manipulation and the shared workspace. This strategy thus allows to benefit from the advantages of a reactive controller to perform a highly complex task in a strongly evolutive environment where various constraints must be taken into account. To validate the approach, a simulation campaign has been conducted. A first step was to evaluate the different prediction models to select the best one for the servoing. Then, the VPC strategy itself was tested, considering different scenarii and using a PR2 model as the considered dual-arm robot. The obtained results have shown its interest and its efficiency to safely perform various positioning tasks in a shared workspace.



(a) Distance between the obstacle and the different links of the right arm - Solid: Distances - Dotted: Minimal allowed distance

(b) Collision distance - Solid: Distances between links - Dotted: Minimal allowed distance

Fig. 16. Collision distance between the right arm and the obstacle, and between the arms

For future works, two main leads can be followed. First, these works only consider a fixed dual-arm system. To go further, it is necessary to mount it on a mobile base and to coordinate the motions of the complete robotic setup. To do so, the proposed control strategy will have to be coupled to our previous works about autonomous navigation in orchards. A second interesting research axis concerns the realization of experimental tests on a real robotic system. To achieve this goal, it will be required to speed up the processing time. The prediction models will have to be selected depending not only on their accuracy but also on their computational time. In this context, designing new models specifically adapted to Graphical Processing Units could offer a nice opportunity. Such aspects are now at the core of our current work in order to perform a complete fruits picking task in a real environment.

References

1. Allibert, G., Courtial, E., Chaumette, F.: Predictive Control for Constrained Image-Based Visual Servoing. IEEE Transactions on Robotics **26**(5), 933–939 (Oct 2010). https://doi.org/10.1109/TRO.2010.2056590
2. Assa, A., Janabi-Sharifi, F.: Robust model predictive control for visual servoing. In: 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems. pp. 2715–2720 (2014)
3. Bergerman, M., Billingsley, J., Reid, J., van Henten, E.: Robotics in agriculture and forestry. In: Springer Handbook of Robotics, pp. 1463–1492. Springer (2016)
4. Cadenat, V., Folio, D., Durand-Petiteville, A.: Comparison of two sequencing techniques to perform a vision-based navigation task in a cluttered environment. Advanced Robotics **26**(5-6), 487–514 (2012)

5. Chaumette, F., Marchand, T.: A redundancy-based iterative approach for avoiding joint limits: application to visual servoing. IEEE Transactions on Robotics and Automation **17**(5), 719–730 (Oct 2001). https://doi.org/10.1109/70.964671, http://ieeexplore.ieee.org/document/964671/
6. Chaumette, F., Hutchinson, S.: Visual servo control. i. basic approaches. IEEE Robotics & Automation Magazine **13**(4), 82–90 (2006)
7. Durand-Petiteville, A., Cadenat, V., Ouadah, N.: A complete sensor-based system to navigate through a cluttered environment. In: 2015 12th International Conference on Informatics in Control, Automation and Robotics (ICINCO). vol. 2, pp. 166–173. IEEE (2015)
8. Durand-Petiteville, A., Le Flecher, E., Cadenat, V., Sentenac, T., Vougioukas, S.: Design of a sensor-based controller performing u-turn to navigate in orchards. In: Proc. Int. Conf. Inform. Control, Automat. Robot. vol. 2, pp. 172–181 (2017)
9. Durand-Petiteville, A., Cadenat, V.: From the general navigation problem to its image based solutions. ViCoMoR 2012 p. 1 (2012)
10. Flécher, E., Durand-Petiteville, A., Cadenat, V., Sentenac, T.: Visual predictive control of robotic arms with overlapping workspace. In: 16th International Conference on Informatics in Control, Automation and Robotics (ICINCO 2019). pp. 130–137 (2019)
11. Foley, J.A., Ramankutty, N., Brauman, K.A., Cassidy, E.S., Gerber, J.S., Johnston, M., Mueller, N.D., O'Connell, C., Ray, D.K., West, P.C., Balzer, C., Bennett, E.M., Carpenter, S.R., Hill, J., Monfreda, C., Polasky, S., Rockstrm, J., Sheehan, J., Siebert, S., Tilman, D., Zaks, D.P.M.: Solutions for a cultivated planet. Nature **478**, 337–342 (Oct 2011)
12. Francois Chaumette: Asservissement visuel. In: La commande des robots manipulateurs, Trait IC2, vol. 3, pp. 105–150. Chap (2002)
13. Garage, W.: Pr2 user manual (2012)
14. Grift, T., Zhang, Q., Kondo, N., Ting, K.C.: A review of automation and robotics for the bio- industry. Journal of Biomechatronics Engineering **1**(1), 19 (2008)
15. Grüne, L., Pannek, J.: Nonlinear model predictive control. In: Nonlinear Model Predictive Control, pp. 45–69. Springer (2017)
16. Hajiloo, A., Keshmiri, M., Xie, W.F., Wang, T.T.: Robust On-Line Model Predictive Control for a Constrained Image Based Visual Servoing. IEEE Transactions on Industrial Electronics pp. 1–1 (2015). https://doi.org/10.1109/TIE.2015.2510505
17. Heshmati-alamdari, S., Karavas, G.K., Eqtami, A., Drossakis, M., Kyriakopoulos, K.J.: Robustness analysis of model predictive control for constrained image-based visual servoing. In: 2014 IEEE International Conference on Robotics and Automation (ICRA). pp. 4469–4474 (May 2014)
18. Kazemi, M., Gupta, K., Mehrandezh, M.: Path-planning for visual servoing: A review and issues. In: Visual Servoing via Advanced Numerical Methods, pp. 189–207. Springer (2010)
19. Ke, F., Li, Z., Xiao, H., Zhang, X.: Visual servoing of constrained mobile robots based on model predictive control. IEEE Transactions on Systems, Man, and Cybernetics: Systems **47**(7), 1428–1438 (July 2017). https://doi.org/10.1109/TSMC.2016.2616486
20. Le Flecher, E., Durand-Petiteville, A., Cadenat, V., Sentenac, T., Vougioukas, S.: Implementation on a harvesting robot of a sensor-based controller performing a u-turn. In: 2017 IEEE International Workshop of Electronics, Control, Measurement, Signals and their Application to Mechatronics (ECMSM). pp. 1–6. IEEE (2017)

21. Mansard, N., Chaumette, F.: Task sequencing for high-level sensor-based control. IEEE Transactions on Robotics **23**(1), 60–72 (Feb 2007). https://doi.org/10.1109/TRO.2006.889487
22. Marchand, E., Chaumette, F., Rizzo, A.: Using the task function approach to avoid robot joint limits and kinematic singularities in visual servoing. In: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems. IROS '96. vol. 3, pp. 1083–1090. IEEE, Osaka, Japan (1996). https://doi.org/10.1109/IROS.1996.568954, http://ieeexplore.ieee.org/document/568954/
23. Mcfadyen, A., Mejias, L., Corke, P., Pradalier, C.: Aircraft collision avoidance using spherical visual predictive control and single point features. In: 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems. pp. 50–56. IEEE (2013)
24. Mezouar, Y., Chaumette, F.: Path planning for robust image-based control. IEEE Transactions on Robotics and Automation **18**(4), 534–549 (Aug 2002). https://doi.org/10.1109/TRA.2002.802218, http://ieeexplore.ieee.org/document/1044366/
25. Van Henten, E., Hemming, J., Van Tuijl, B., Kornet, J., Bontsema, J.: Collision-free Motion Planning for a Cucumber Picking Robot. Biosystems Engineering **86**(2), 135–144 (Oct 2003). https://doi.org/10.1016/S1537-5110(03)00133-8
26. Vougioukas, S.G., Arikapudi, R., Munic, J.: A Study of Fruit Reachability in Orchard Trees by Linear-Only Motion. IFAC-PapersOnLine **49**(16), 277–280 (2016). https://doi.org/10.1016/j.ifacol.2016.10.051
27. Wang, T., Xie, W., Liu, G., Zhao, Y.: Quasi-min-max model predictive control for image-based visual servoing. In: 2012 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM). pp. 98–103 (July 2012). https://doi.org/10.1109/AIM.2012.6265955
28. Zhao, Y., Gong, L., Huang, Y., Liu, C.: A review of key techniques of vision-based control for harvesting robot. Computers and Electronics in Agriculture **127**, 311–323 (Sep 2016). https://doi.org/10.1016/j.compag.2016.06.022
29. Zhao, Y., Gong, L., Liu, C., Huang, Y.: Dual-arm Robot Design and Testing for Harvesting Tomato in Greenhouse. IFAC-PapersOnLine **49**(16), 161–165 (2016). https://doi.org/10.1016/j.ifacol.2016.10.030