Planning Actions by Interactive Movement Primitives: pushing occluding pieces to pick a ripe fruit

Sariah Mghames, Marc Hanheide and Amir Ghalamzan E.

Abstract—While many efforts are currently devoted by research bodies to investigate in robot harvesting, challenges related to picking fruits from clusters are still considered an open issue which can limit the operation success. On the other hand, existing planning frameworks for robotic manipulation in cluttered and uncertain environment are getting more and more attention for their ability to deal with physics-based strategies to free the robot path to a goal object. However, those approaches are either computationally expensive and/or designed for 2-D occlusion scenes. Consequently, they are not readily applicable to the complex 3-D geometry of fruits in clusters. In this work, we present a path planning algorithm for pushing occluding fruits to reach-and-pick a ripe one. Hence, we propose an Interactive Probabilistic Movement Primitives (I-ProMP) which is computationally efficient and is readily used for 3-D problems. We demonstrate the efficiency of our approach with pushing unripe strawberries in a simulated polytunnel. Our experimental results confirm I-ProMP successfully pushes table top grown strawberries and reaches a ripe one.

I. INTRODUCTION

State-of-the-art path planning algorithms do not tackle the problem of fast path generation for a robotic manipulator in a 3-D cluttered scene with connected objects [1]–[6]. In this work, we propose an Interactive Movement Primitives (IMP) strategy, that allows us to quickly plan simple quasi-static pushing movements, e.g. for fruit picking [7] where the motion planning must readily generalise to different configurations of fruits in clusters (fig. 1).

Labour shortage is a major challenge for many sectors including agriculture. In the UK alone, the soft fruit sector uses 29,000 seasonal pickers to produce over 160,000 tons of fruit every year [7]. Only strawberry harvesting cost is more than 60% of the total production cost. Bringing robotic arms to the field is a response to this global challenge of labour shortage [8]. However, precise, reliable and fast motion planning is one of the key bottlenecks of a robotic fruit picker [7]. A sophisticated robotic picking technology (fig. 2) is only capable of successfully picking isolated strawberries whereas many of the strawberries are grown in clusters [9].

An increasing number of robotic harvesting technologies are nowadays presented. Schuetz et al. in [10] formulated the harvesting problem as a static optimal control problem relying on an initially generated heuristic trajectory. In their work, the authors generate an optimal harvesting trajectory that minimises the collision and dynamical costs. In [11], the authors presented an energy optimal combined with an artificial potential field approach to formalise the problem of a collision-free path-planning harvesting 6-DoFs robot. In [7], the authors propose an active obstacle-separation path planning strategy for picking fruits in clusters inspired by human pickers who usually use their hands to push and separate surrounding obstacles during picking. In the latter work, the authors adapt a pushing action on the obstacle fruits before reaching the target one. The pushing action results mechanically from the move in a designed direction, thanks to a genuine design of a fingers-like gripper. Although a pushing action is generated along the path to the target, it doesn’t rely on the cluster physics, and hence is considered heuristic and limits the picking success rate. In addition, the approach in [7] lacks consideration of different kind of occlusions (e.g those coming from stems) which may result in grasp failure due to components coupling, and also, the strategy doesn’t consider combined cases (e.g top obstacles on symmetrical side of the target) which can leave the proposed planning strategy without solution.

In addition to picking technology, many other agriculture robotics are researched including, but not limited to, weed detection and removal [12], crops growth monitoring through aerial robots (e.g. using quadcopters) [13], fruits [14] and plants [15], [16] detection and tracking, mobile...
robot navigation and mapping [17], [18]. Another agri-
robotics area is focusing more on robot kinematics and
manipulation trying to find a suitable and efficient gripper
design. Some grippers are developed to achieve stable and
soft contacts [19] with the fruits and others are developed
based on a scissor-like concept [9].

A human may push/move objects to reach-and-pick a
fruit or reach-and-grasp an object in a toolbox or fridge.
Some previous studies researched some of such problems
in robotic context. For instance, [3]–[6] consider problems
limited to 2-D problem of objects rearrangement on a
flat surface, e.g. in a fridge or on a shelf, to reach and
grasp the desired object in a cluttered scene. Although
the approach in [3] is computationally efficient for no-
uncertainty case, it is proposed for 2-D occlusion scene. In
examples [4]–[6], computationally expensive approaches,
such as physics-based trajectory optimisation, are success-
fully used. However, these approaches require a long
computation time for planning the movements (e.g. in [5]
an average computation time for 2 objects in a refrigerator
is reported for ∼10 [s]).

In other real-world 3-D examples, e.g. picking fruits,
the interactions of the objects with its environments may
not involve complex computations, e.g. friction between
object and the flat surface. As such, those approaches are
not useful because they are designed for 2-D problems
and need long time for planning and performing pushing
movements.

In contrast to optimisation based approaches, robot
learning from demonstration (LfD) approaches, e.g. [20],
have been successfully developed to minimise the planning
time. For instance, Dynamic Movement Primitives (DMP)
are used to generate and adapt the robot trajectory in real-
time [21] where they can also be used to avoid colli-
sion [22]. Probabilistic Movement Primitives (ProMP) is
also an LfD approach which features interesting properties
useful for our problem [23]–[24]. For example, Maede
et al. [25] proposed an interaction learning method for
collaborative and assistive robots based on probabilistic
movement primitives. With ProMPs, we are able to en-
code variability and uncertainty in the movements and to
derive new operations which are essential for implementing
modulation of a movement, coupling, co-activation and
temporal scaling. Shyam et al. [26] proposed a probabilis-
tic primitive based optimisation technique to generate smooth
and fast trajectories. Their approach relies on the Covariant
Hamiltonian optimisation framework for motion planning
with obstacle avoidance constraints initialized with a prob-
abilistic primitive.

In this paper, we extend ProMP [26] and propose
an Interactive-ProMP (I-ProMP) planning the pushing of
unripe strawberries. Our contribution is manifold as follows:
(i) we present primitive cluster types in challenging
strawberry picking problem which defines different con-
figurations of strawberries; (ii) we present I-ProMP which
efficiently generates, in 0.19[s] mean computation time
and 0.0022 standard deviation, movements necessary for
pushing strawberries occluding the robot’s way to a ripe
strawberry; (iii) we develop a simulation environment in
Gazebo 8.0 which allows us to test our I-ProMP. Hence,
we test I-ProMP in the developed simulation environment
which illustrates our approach successfully performs the
pushing movements and reaches the occluded ripe straw-
berry in different configurations.

The rest of the paper is divided into the following:
section [II] defines the problem and the current main chal-
lenge in fruit harvesting, section [III] presents the approach
we followed to tackle the problem, section [IV] illus-
trates experimental results reported on a simulated field and
section [V] concludes with future works.

II. PROBLEM FORMULATION

Robotic fruit picking is an interesting motion/path plan-
ning problem. For instance, a ripe fruit to be picked may
be located among leaves and unripe fruits where the robot
can neither fully observe the fruit nor plan a collision-free
robotic arm to reach-and-pick the ripe one. Cluster config-
urations of fruits are determined by the fruit variety which
may result in varying number of roots and nodes. The
sophisticated end-effector design for strawberry picking [7]
fails to pick strawberries in clusters (Fig. 2b) because the ripe strawberries may be occluded by unripe ones. In this paper, we adopt Interactive Probabilistic Movement Primitives (I-ProMP).

We identified a few primitive fruit configurations in a cluster illustrated in fig. 3 and fig. 11, which resemble most of the real strawberry clusters. Other cluster types may be formed by combining the primitive configurations:

(i) clusters with isolated and non-occluded components (fig. 3a) – ripe target above and below and at the same level of unripe neighbours, respectively from left to right; (ii) clusters with connected components – ripe strawberry at the same level, above and below the neighbour strawberries (fig. 3b), respectively from left to right; (iii) a cluster with an occluded target where the target and occluding strawberry have different nodes (fig. 3c); (iv) a cluster with occluded target where the target and occluding strawberry (fig. 3d).

We simulate all the cluster configurations in Gazebo 8.0 simulation framework. Each stem is equipped with a 3-axes hinge at its root. A discussion for handling each case is elaborated later on in section IV.

III. PROPOSED APPROACH

In this section, we present our proposed Interactive Probabilistic Movement Primitives (I-ProMP). We used cubic radial basis functions for generating sample trajectories, called demonstrations, by \( \varphi(x) = \|x - c\|^3 \), where \( x \) is the input variable and \( c \) is a fixed point, called the center of the function. We generated 10 sample nominal trajectories (fig. 4) with the same initial and 10 different end-points. For each nominal trajectory, we have 10 samples where their end-points are randomly sampled with the mean equivalent to the nominal goal point and standard deviation of 10^{-3}. We consider completion time for the demonstrated trajectories to be \( T = 1 \) [s].

We model a movement execution as a trajectory \( \xi = \{X_t\}_{t=0}, \ldots, T \), defined by the end-effector pose \( X_t \) over time. A ProMP model \( \psi \) describes multiple ways to execute a movement, which naturally leads to a probability distribution over trajectories. The latter can be represented by a deterministic function of weights \( \omega \) and phase variable \( z(t) \), as follows:

\[
X_t = \psi^T \omega + \varepsilon_t
\]

where \( \psi_t \in \mathbb{R}^{n \times 3} \) is a basis matrix, \( \varepsilon \) is a zero mean Gaussian random variable with variance \( \Sigma_z \). We choose \( k \) gaussian basis functions which have been shown to be good enough for non-periodic movements,

\[
\psi^T_k = \exp\left(-\frac{(z_t - c_t)^2}{2h}\right)
\]

where \( z_t \) is a time-dependent phase variable, \( c_k \) is the center of the \( k \)th basis function and \( h \) is the width of the basis. Basis functions in Eq. 2 are normalised by \( \Sigma \psi_j(z) \).

1) ProMP trained by demonstrations: In order to learn a movement primitive with properties similar to the generated demonstrations, we learn weight parameters using an extension of the maximum a-posteriori probability (MAP) estimate \( \hat{\theta} \), as follows

\[
\omega = (\lambda I + \psi^T \psi)^{-1} \psi^T X
\]

where \( \lambda \) is a regularisation term to avoid overfitting in the original optimisation objective \( \psi \theta \). The probability of observing a trajectory \( X \) given the parameter vector \( \theta = \{\mu_\omega, \Sigma_\omega\} \) is given by the marginal distribution

\[
p(X_t; \theta) = \mathcal{N}(X_t | \mu_\omega, \Sigma_\omega)
\]

where \( \mu_\omega \) and \( \Sigma_\omega \) are the mean and variance of the weight vector respectively.

2) ProMP conditioning at a goal neighborhood: We consider a scenario in which the robot has a camera looking at the table from the side view localising a ripe strawberry at \( p_{rs}(t) = [x_1, x_2, x_3] \). The picking robot first moves to the bottom of the ripe fruit, \( p_{bs}(t) = [x_1, x_2, x_3 - 0.1 \text{ m}] \), which is captured by the first camera, to get a better view of the cluster with a camera-in-hand. Then, the robot performs the push movements to open occlusions and reach the target strawberry. We consider the picking actions are cyclic, i.e. after picking strawberry \( i \), the robot plans the movements to pick strawberry \( i + 1 \) and so on. The picking head of the robot is equipped with a punnet; so, the robot directly picks the strawberries into the punnet. As such, target picking position at time \( t \) becomes initial position for planning the next picking movements. At time \( t \), we condition the ProMP at \( p_{bs} \) with time \( t + 0.85 \) [s] and at \( p_{rs} \) with time \( t + T \) [s]. In addition, we synthesise a systematic strategy to condition the primitive at selective neighbor fruits, as a first attempt to create a pushing action in an occluded scene.

Fig. 5 compares the effect of the number of basis functions on the regeneration performance of the learnt ProMP. This figure shows that a ProMP with larger number of basis functions can capture/generate higher nonlinear behaviours.

We distinguish an approaching trajectory, which has the least non-linearity, and a pushing trajectory, which may be
Fig. 5. Learnt ProMP: (a) with number of basis \( \psi = 4 \), bandwidth \( h = 1 \) and \( T = 1 \) [s]; (b) with \( \psi = 10 \), \( h = 1 \) and \( T = 1 \) [s]. As it these figures show, ProMP with larger number of basis functions can capture more nonlinearity of the demonstrated trajectories.

Fig. 6. (a) One learnt ProMP with number of basis functions \( \psi = 20 \), bandwidth \( h = 1 \) (b) \( \psi = 10 \), \( h = 1 \) and (c) \( \psi = 4 \), \( h = 1 \), all with conditioning time vector \( T_c = [0, 0.85, 1, 1.3, 1.6, 2] \) [s]. (d) One learnt ProMP with \( \psi = 4 \) and conditioning time \( T_c = [0, 1.2, 1.4, 1.6, 1.8, 2] \) [s], (e) Two learnt ProMPs with \( \psi_1 = 4 \) and \( \psi_2 = 4 \), (f) Two learnt ProMPs with \( \psi_1 = 4 \) and \( \psi_2 = 5 \). \( \psi_2 \) is associated with ProMP1 learnt from data spreading \( t = 0 \) to \( t = 0.85 \) while \( \psi_2 \) is associated with ProMP2 learnt from data spreading \( t_1 = 0.85 \) to \( T = 1 \) [s].

highly non-linear. As such, we consider two ProMPs which represent different level of non-linearities with respect to the phase variable input. The first ProMP, namely \( MP_1 \), is used to generate reach-to-pick, whereas the second ProMP, namely \( MP_2 \), is used to generate push-to-pick, as follows:

\[
\begin{align*}
\text{MP}_1[\psi_1(z_t)], & \quad 0.0 \leq t \leq 0.85, \\
\text{MP}_2[\psi_2(z_t)], & \quad 0.85 \leq t \leq 1,
\end{align*}
\]

where, \( MP_1 \) and \( MP_2 \) have \( k = 4 \) and \( k = 5 \) basis functions, respectively. Comparing the generated trajectories using \( MP_1 \) and \( MP_2 \) models for generating reach-to-pick and push-to-pick trajectories (fig. 6 e-f) versus using just one ProMP over completion time \( T = 1s \) (fig. 5) and one ProMP over completion time \( T = 2s \) (fig. 6 a-d), shows the superiority of two ProMPs in our specific example. Figure 5a shows that, with a single primitive we are not able to achieve zero variance ProMP at the conditioned neighbor fruits (green spheres) while holding the time duration of the demonstrations, i.e. \( T = 1s \). On the other hand, for an increased number of basis functions (e.g \( \psi = 10 \), as in fig. 5b) a non-linear behavior is induced along the whole trajectory for a 4 points conditioning requirement (green spheres), in addition to the final (red sphere) and initial condition (yellow sphere). Hence, we double the total trajectory duration to \( T = 2s \) and compare the use of single MP with different number of basis: \( \psi = 20 \) (a), \( \psi = 10 \) (b) and \( \psi = 4 \) (c), all with the same discretized conditioning time \( T_c = [0, 0.85, 1, 1.3, 1.6, 2] \) [s] going from \( t_0 = 0s \) to \( T = 2s \) (red). The better case turns out to be the one with \( \psi = 10 \). At this point, we compare this case with a similar one, but with a different discretized time duration \( T_c = [0, 1.2, 1.4, 1.6, 1.8, 2] \) [s] (fig. 6c). It turns out that the improvement in smoothness we got in the second phase of the trajectory \( (T = 1.2s \rightarrow T = 2s) \) results in larger variations in the trajectory in the approaching phase \( (T = 0 \rightarrow T = 1.2s) \). As a consequence, we test the learning phase on two different time zones separately while keeping the duration \( T = 2s \). One ProMP is learnt from demonstrations lasting for \( t_1 = 0.85s \) and the other is learnt for the remaining time duration \( (t_1 = 0.85s \rightarrow T = 1s) \).

Figure 6c shows the resulting ProMP with \( \psi_{i,2} = 4 \) while 6f shows the ProMP generated with \( \psi_1 = 4 \) and \( \psi_2 = 5 \) for the consecutively learnt MP1.

Based on the results obtained in fig. 6f that we choose to adopt in this work, it is worth noting that this approach results close to the strategic method presented in the work of Stefan Schaaf’s Lab [28] for imitation learning, where many narrow basis functions are placed where the function is highly nonlinear whereas fewer and wider basis functions are placed in linear areas.

Given the desired end-effector observation \( x_t^* = [X_t^*, \Sigma_t^*] \) with a desired variance \( \Sigma_t^* \), we can get the Gaussian conditional distribution \( p(\omega_t^* | x_t^*) \) for the updated weight vector \( \omega_t^* \) with mean and variance the following,

\[
\begin{align*}
\mu_{\omega_t^*} &= \mu_\omega + \Sigma_\omega \psi_t (\psi_t^T \Sigma_\omega \psi_t)^{-1} (X_t^* - \psi_t^T \mu_\omega), \\
\Sigma_{\omega_t^*} &= \Sigma_\omega - \Sigma_\omega \psi_t (\psi_t^T \Sigma_\omega \psi_t)^{-1} \psi_t^T \Sigma_\omega.
\end{align*}
\]

In a second phase, an online decision-making on the neighbored fruits to condition upon is made by a planner presented in the following, based on the criteria \( i \cdot d_{ng} \leq r_{max} \), where \( d_{ng} \) is the euclidean distance between a neighbor fruit and the target one, projected on unit vector along the table top axis \( i = [1, 0, 0] \). \( r_{max} = D_d/2 \) is the maximum gripper opening radius, 3cm.

A. Interactive-ProMP

In the following section, we present a Stochastically-driven Interactive Planner (SIP) which implements a probabilistic physics-based pushing strategy to swallow a target fruit from an 3-D cluster.

With the ProMP generated above (Eqs. 4-5) and illustrated in fig. 7 two problems can still arise depending on the cluster configuration in scene:

- The ProMP may cross the stem or the target fruit before being able to swallow it. Consequently, a
the conditioned movement primitive that features the swallow success rate, we present hereafter a modified version of the conditioned movement primitive. In order to reduce the aforementioned risks and increase the swallow success rate, we present hereafter a modified version of the conditioned movement primitive. In order to reduce the aforementioned risks and increase the swallow success rate, we present hereafter a modified version of the conditioned movement primitive.

Fig. 7. Sequential probabilistic primitives generated from an initial state (IC) to Goal1 fruit (green sphere) while conditioning on its neighbored fruits (red spheres) as a first attempt to generate a pushing action, then from Goal1 to Goal2 followed by Goal2 to Goal3. The 3 primitives are illustrated for 3 types of clusters shown in fig. 3a. Two task space learnt probabilistic movement primitives are conditioned at a distance 10cm (cyan spheres) below the goal for a close fruit detection.

Fig. 8. (a) Mock-up setup with SCARA arm, a finger-like gripper, an RGB-D camera and a fake strawberry cluster. (b) Finger-like gripper [7] integrated with a scissor and 3 infra-red sensors to localize the target inside.

The SIP algorithm: the SIP is presented in two stages. Algorithm 1 takes as input a set of ripe \( \mathbf{G} \) and unripe \( \widehat{\mathbf{G}} \) fruits and output the position of the pushable objects selected at time \( t \) (\( P_P^t \)), their orientation at time \( t \) (\( P_P^t \)), estimated by the stem orientation (assumed given in this work), their updated position at \( t \) (\( P_P^t \)). At first, the algorithm selects a target \( g_t \), generates a cluster \( C_g^0 \) associated to \( g_t \) using the radius nearest neighbors technique (RNN) where \( r = 0.05m \) is the chosen cluster radius. The clustered points are then divided into bottom, plane and top subsets, \( S_d, S_p, S_t \) respectively, reduced by the criteria \( (i \cdot d_{ng}) \leq r_{max} \). \( S_d \) is ordered incrementally along axis \( k \) \( k = [0, 0, 1] \) (operation 6), then optimized for the number of elements (operation 7). The optimization takes into account the points in the subset that are at quasi-equal level and selects the one (\( S_d^* \)) with smaller \( (i \cdot d_{ng}) \) value, in case their stems don’t occlude the target, else (e.g fig. 5d) it chooses the one with stem less inclined with respect to the target because it needs less effort to push it. The latter fact is not taken as a showcase in this work, related results will be reported in our future work when an optimization problem formulation will be elaborated for that purpose. Given the stem orientation \( u_{stem} \), the algorithm computes a stem length estimation, \( L_{stem} \), for every element in the optimized subset, i.e \( n_d^* \). After getting the intersection of \( u_{stem} \) with the table top plane (operation 9), then gets the inclination angle \( \theta_0 \), computes the total inclination \( \theta \) needed to free the maximum gripper opening (operation 12), and gets the minimum displacement required (\( s \)) of the pushable object to free the path to the target. Two possible pushing directions are proposed, \( v_1 \) and \( v_2 \), each normal to the stem direction \( u_{stem} \) (assumed to be vertical, hence aligned with \( k \), only for the case of determining the pushing direction \( u_p \)). If the algorithm finds other elements from \( S_p \) at same level as \( n_d^* \) (e.g fig. 5d), it pushes \( n_d^* \) along \( v_2 \) (i.e normal to table top), else, it pushes \( n_d^* \) along \( v_1 \) (i.e parallel to table top). The updated \( n_d^* \) coordinates (i.e \( P_P^t \)) are computed by projecting the displacement vector \( s \) on

Algorithm 1 The 3-D Pushing Strategy

\[ \text{Input: } \mathbf{G} = \{\text{matured fruits}\}, \widehat{\mathbf{G}} = \{\text{non-matured fruits}\} \]
\[ \text{Output: } P_P^0, P_P^1, P_P^2 \]

1: \text{procedure PUSHING NEAREST NEIGHBORS}
2: \hline
3: \hline
4: \hline
5: \hline
6: \hline
7: \hline
8: \hline
9: \hline
10: \hline
11: \hline
12: \hline
13: \hline
14: \hline
15: \hline
16: \hline
17: \hline
18: \hline

Algorithm 2 Interactive ProMP generation

```
Input: g_{t-1}, g_t, P^s_p, P^t_p
Output: Interactive ProMP

procedure ProMP
2: P_{cond} = \{g_{t-1}, P^s_p, P^t_p, g_t\}
\omega_{ML} \leftarrow Eq.(3)
4: \mu_0, \Sigma_0
for wpt \in G_{cond} do
6: x_d, \Sigma_d
Update: \mu_0, \Sigma_0 \leftarrow Eq.(6), (7)
8: end for
end procedure
```

the pushing direction \( \mathbf{u}_p \) (operation 16).

Algorithm 2 takes as input the latest ripe goal \( g_{t-1} \), a new ripe goal \( g_t \), \( P^s_p \) and \( P^t_p \). It then conditions the learnt ProMPs at the input set (operation 5) and updates the parameters of the weight distribution (operation 7).

**Discussion:** Clusters of type Goal2 and Goal3 in fig. 7 are not considered at this stage of work by SIP, hence \( S_p \) and \( S_t \) are eliminated in algorithm 1. In a future work, we will benefit from the genuine finger-like design of the gripper in use (shown in fig. 3d) to test the grasp of corresponding scenarios with pushing actions generated kinematically by an incremental opening and closing of the active fingers.

IV. EXPERIMENTS AND RESULTS

Given a finger-like gripper design [9] that we adopt in our application, we consider that configurations of type 3a and 3b can be targeted with only a primitive-based planner applied on the ripe target. A pushing action will result mechanically from path following. This hypothesis is tested for the case in which the target is above neighbors and results are reported in fig. 10 where fig. 10a represents a ProMP generated by the planner and fig. 10b represents the simulated end-effector trajectory (in blue), unripe fruits trajectory (in green) and target fruit trajectory (in red).

For the case of fig. 3d as discussed in section III this type of cluster gets optimized for the number of neighbors to push and gets a pushing direction along \( \mathbf{v}_2 \), i.e normal to table top. In the following, we present results related to the configuration in fig. 3c and other two descendants.

A. Hardware Platform

Two SCARA arms are mounted on Thorvald mobile robot [29] for fruit harvesting. Each arm is a 3-DoFs PRR serial chain and consists of a cable-driven fingers-like end-effector whose role is to swallow a fruit strawberry, center it and finally cut it with an internal scissor. For a detailed description of the gripper the reader can refer to [9]. Whenever the goal fruit is detected by 3 integrated infra-red sensors, the end-effector \( X_c \) position is increased by 2\text{cm} to cut the stem with the scissor. As the growing season starts in June [8], we only present our results in a simulated polytunnel.

![Fig. 9. Field simulation on the SCARA arm mounted on a mobile base, the Thorvald robot (b), navigating in polytunnels (a)](image)

B. Simulations and Results

We developed a simulation environment in Gazebo 8.0, reported in fig. 2a and consisting of a polytunnel with parallel table tops. In addition to the configurations shown in fig. 3 we built complex clusters (figs. 11b - 11c) out of the one shown in fig. 11a and attached them to the table top. We sent Thorvald in the polytunnel and command the arm to follow a joint trajectory generated from the SPI planner. For the 3 simulated clusters, i.e figs. 11a - 11c the SPI planner generates an interactive ProMP in the robot Cartesian space, shown in figs. 11d - 11f with a blue color, respectively. Through the robot inverse kinematics, joint trajectories, constrained to joint limits, are passed to the SCARA arm and monitored by an effort controller exposed to a follow joint trajectory action interface. We record the actual joint trajectories from the simulation and, with the use of the robot forward kinematics, we report in figs. 11g - 11i the actual end-effector trajectory and the trajectory of each strawberry in the scene.

**Discussion:** In figs. 11a - 11i every light pink sphere (without green stem) under or above the goal represents the initial pose of an unripe neighbor, accounted for with a horizontal shift in position. The latter shifted position is added to the conditioned set points and illustrated with a short stem. The shift accounts for an amount equal to \( r_{max}^g + r_{max}^f \) where \( r_{max}^g \) is an estimated maximum fruit radius; this is to align the end-effector with the pushing direction \( \mathbf{u}_p \). We can see that the planner doesn’t pass through the unripe fruit above the target in fig. 11i while the target sphere has

\[ P: \text{prismatic}, \ R: \text{revolute} \]
Fig. 11. Probabilistic primitive-based pushing strategy to swallow a soft fruit from complex clusters: (a) cluster with target fruit (red) occluded by one stiff inclined stem and an unripe fruit (green), (b) cluster with target occluded by two stiff, unripe stem-fruit system, each from one side, (c) cluster with target fruit occluded by two stiff stem-fruit system, one from above and another one from below, (d-f) Interactive ProMP generated for the clusters (a-c): the target is conditioned at a point directly below it (representing the target radius, 1.5cm, plus a margin of 0.1cm). The pushed unripe fruit is connected to the shifted pose with an inclined green stem. The legend in (d) applies to (e-f) while legend in (g) applies to (h-i) too. (g-i) show the actual end-effector trajectory, extracted from the simulation environment. Also, they illustrate the goal fruit trajectory and the neighbor unripe fruit trajectories.

In figs. 11g - 11i, it can be noticed that the actual end-effector trajectory passes through the unripe fruits, and reaches to the target (quasi-static position over time). We note that, the generated interactive primitive is not passed with multiple way-points to the robot joints before the camera position (cyan sphere), hence a larger difference between the generated one and the simulated one exists before reaching the camera conditioning position. In table I, a metric is retrieved to prove the occurrence of the pushing action. We consider a contact has occurred if the minimum distance traveled by the gripper frame with respect to a frame attached to fruit center, at the fruit altitude is:

$$d_{f,ee}^\text{min} \leq (r_{f}^\text{max} + r_{g}^\text{max}) = 4.5\text{cm}.$$ For cases of $C_1$-neighbor and neighbor of configuration 11i it results that the contact has occurred with the conical gripper surface below the frame attached to its vertex. This work is accompanied with an attached video reporting the simulation of each case scenario studied.

**V. Conclusion**

In this work, we presented an interactive primitive-based planning strategy that features pushing actions in complex clusters. Although it can be generalized to different applications, the proposed approach targets a specific application, the one of robotic harvesting, and hence tackles the problem of picking occluded fruits. Occlusion results normally from the variety of grown clusters. In order to generate different degrees of non-linearity in the system behavior, the planner learns two movement primitives from demonstrations, conditions the resulting primitive to pass
through selective neighbors (movable obstacles), reasons then on the pushing direction of each of them and finally augments them with an updated pose. We tested our approach on different complex cluster configurations in a simulated polytunnel using a SCARA robotic arm mounted on a mobile base. The strategy succeeded to reach the target in the different scenarios selected. As part of future work, the pushing planner will be tested on real field at the beginning of the coming strawberry season with an online closed loop feedback and primitive update. In addition, an optimization approach will be developed to take into consideration fixed obstacles and the minimum number of fruits to push.

ACKNOWLEDGEMENT

The authors would like to thank SAGA Robotics Ltd and more specifically Callum Small and Kristoffer Kskarsgard, for their support and contribution on the hardware mock-up platform (Thorvald Harvester).

REFERENCES

[1] N. Ratliff, M. Zucker, J. A. Bagnell, and S. Srinivasa, “Chomp: Gradient optimization techniques for efficient motion planning,” 2009.

[2] M. Kalakrishnan, S. Chitta, E. Theodorou, P. Pastor, and S. Schaal, “Stomp: Stochastic trajectory optimization for motion planning,” in 2011 IEEE international conference on robotics and automation, IEEE, 2011, pp. 4569–4574.

[3] M. R. Dogar and S. S. Srinivasa, “A planning framework for nonprehensile manipulation under clutter and uncertainty,” Autonomous Robots, vol. 33, no. 3, pp. 217–236, 2012.

[4] W. C. Aghob and M. R. Dogar, “Real-time online re-planning for grasping under clutter and uncertainty,” in 2018 IEEE-RAS 18th International Conference on Humanoid Robots (Humanoids). IEEE, 2018, pp. 1–8.

[5] N. Kitaev, I. Mordatch, S. Patil, and P. Abbeel, “Physics-based trajectory optimization for grasping in cluttered environments,” in 2015 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2015, pp. 3102–3109.

[6] J. E. King, J. A. Haustein, S. S. Srinivasa, and T. Asfour, “Nonprehensile whole arm rearrangement planning on physics manifolds,” in 2015 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2015, pp. 2508–2515.

[7] Y. Xiong, Y. Ge, L. Grimstad, and P. J. From, “An autonomous strawberry-harvesting robot: Design, development, integration, and field evaluation,” Journal of Field Robotics, 2019.

[8] “Sample costs to produce and harvest strawberries,” [Online]. Available: https://coststudfiles.ucdavis.edu, accessed: 23-02-2020.

[9] Y. Xiong, C. Peng, L. Grimstad, P. J. From, and V. Isler, “Development and field evaluation of a strawberry harvesting robot with a cable-driven gripper,” Computers and electronics in agriculture, vol. 157, pp. 392–402, 2019.

[10] C. Schuetz, J. Baur, J. Pfaff, T. Buschmann, and H. Ulbrich, “Evaluation of a direct optimization method for trajectory planning of a 9-dof redundant fruit-picking manipulator,” in 2015 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2015, pp. 2660–2666.

[11] L. Luo, H. Wen, Q. Lu, H. Huang, W. Chen, X. Zou, and C. Wang, “Collision-free path-planning for six-dof serial harvesting robot based on energy optimal and artificial potential field,” Complexity, vol. 2018, 2018.

[12] A. Tellaiche, X. P. Burgos-Artizzu, G. Pajares, and A. Ribeiro, “A vision-based method for weeds identification through the bayesian decision theory,” Pattern Recognition, vol. 41, no. 2, pp. 521–530, 2008.

[13] J. Roldán, P. García-Aunon, M. Garzón, J. de León, J. del Cerro, and A. Barrientos, “Heterogeneous multi-robot system for mapping environmental variables of greenhouses,” Sensors, vol. 16, no. 7, p. 1018, 2016.

[14] R. Kirk, G. Cierniak, and M. Mangan, “L*a*b* fruits: A rapid and robust outdoor fruit detection system combining bio-inspired features with one-stage deep learning networks,” Sensors, vol. 20, no. 1, p. 275, 2020.

[15] K. Yamamoto, W. Guo, and S. Ninomiya, “Node detection and internode length estimation of tomato seedlings based on image analysis and machine learning,” Sensors, vol. 16, no. 7, p. 1044, 2016.

[16] A. Tabb and H. Medeiros, “A robotic vision system to measure tree traits,” in 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2017, pp. 6005–6012.

[17] L. Nardi and C. Stachniss, “Actively improving robot navigation on different terrains using gaussian process mixture models,” in 2019 International Conference on Robotics and Automation (ICRA). IEEE, 2019, pp. 4104–4110.

[18] J. P. Fentanes, I. Gould, T. Duckett, S. Pearson, and G. Cierniak, “3-d soil compaction mapping through kriging-based exploration with a mobile robot,” IEEE Robotics and Automation Letters, vol. 3, no. 4, pp. 3066–3072, 2018.

[19] K. Elgeneidy, P. Lightbody, S. Pearson, and G. Neumann, “Characterizing 3d-printed soft fin ray robotic fingers with layer jamming capability for delicate grasping,” in 2019 2nd IEEE International Conference on Soft Robotics (RoboSoft). IEEE, 2019, pp. 143–148.

[20] A. Ghaliomzian and M. Ragaglia, “Robot learning from demonstrations: Emulation learning in environments with moving obstacles,” Robotics and Autonomous Systems, vol. 101, pp. 45–56, 2018.

[21] S. Schaal, “Dynamic movement primitives—a framework for motor control in humans and humanoid robotics,” in Adaptive motion of animals and machines. Springer, 2006, pp. 261–280.

[22] D.-H. Park, H. Hoffmann, P. Pastor, and S. Schaal, “Movement reproduction and obstacle avoidance with dynamic movement primitives and potential fields,” in Humanoids 2008-8th IEEE-RAS International Conference on Humanoid Robots. IEEE, 2008, pp. 91–98.

[23] A. Paraschos, C. Daniel, J. R. Peters, and G. Neumann, “Probabilistic movement primitives,” in Advances in neural information processing systems, 2013, pp. 2616–2624.

[24] A. Paraschos, C. Daniel, J. Peters, and G. Neumann, “Using probabilistic movement primitives in robotics,” Autonomous Robots, vol. 42, no. 3, pp. 529–551, 2018.

[25] G. J. Maeda, G. Neumann, M. Ewerton, R. Lioutikov, O. Kroemer, and J. Peters, “Probabilistic movement primitives for coordination of multiple human–robot collaborative tasks,” Autonomous Robots, vol. 41, no. 3, pp. 593–612, 2017.

[26] R. Shyam, P. Lightbody, G. Das, P. Liu, S. Gomez-Gonzalez, and G. Neumann, “Improving local trajectory optimisation using probabilistic movement primitives.” IEEE/RSJ, 2019.

| $h_{ee}^{max} (cm)$ | $h_{ee}^{min} (cm)$ |
|----------------|-----------------|
| $g_1$ Neighbor | 2.12 | 2.12 |
| Neighbor | 4.95 | 4.95 |
| Neighbor | 1.22 | 1.22 |
| $g_2$ Neighbor | 1.309 | 1.309 |
| Neighbor | 2.784 | 5.107 |
| $g_3$ Neighbor | 0.7 | 0.75 |
| Neighbor | 0.6 | 0.96 |
| Neighbor | 1.6 | 5.98 |
| $g_4$ Neighbor | 2.46 | 2.49 |
| Neighbor | 6.1 | 8.3 |
[27] C. M. Bishop, *Pattern recognition and machine learning*. Springer, 2006.

[28] T. DeWolf, “Dynamic movement primitives part 1: The basics,” URL: http://studywolf.wordpress.com/2013/11/16/dynamic-movement-primitives-part-1-the-basics, 2014.

[29] L. Grimstad and P. J. From, “Software components of the thorvald ii modular robot,” 2018.