APPLE-E: A Compliant Apple Picking Robot

Udayan Bulchandani
Harvard University, Boston MA 02163, USA

***Abstract*—This work explores some of the challenges involved in building a real-world fruit harvesting system. The problem setting consists of a floating 7-DoF arm and a procedurally generated apple tree, modelled as branching rigid links coupled by compliant spherical joints. For motion planning through dense deformable branch structures, a two-pass strategy is developed. The first pass generates a straight-line trajectory to the target fruit, while the second pass adds curvature to trajectory segments intended to deform branches in a way that allows the end-effector to reach its goal. To avoid applying excessive forces to the tree or crop, APPLE-E uses model predictive control during trajectory execution with the goal of minimizing contact forces experienced by the arm's surface. To measure these contact forces, an approximation of visuotactile sensing is implemented by placing many depth cameras throughout the arm's interior volume. These depth cameras detect any geometries that intersect with the arm's surface, from which contact force can be estimated.**

I. Introduction

Many attempts have been made to build robots capable of performing agricultural harvesting tasks, but none come close to matching human capabilities across different crop families. Agriculture presents a challenging environment for robots to operate in, with complicated obstacles. Robots must navigate uneven terrain and pass through foliage without damaging crops, though some allowable deformation of foliage is often required to complete a particular harvesting task. The task itself may require fine manipulation of an object that is easily damaged if applied forces are too great. This problem presents interesting challenges across hardware design, perception, motion planning and control.

This work considers the problem of apple-picking with some simplifying assumptions. First, it is assumed that a floating 7-DoF arm can be placed in a location from which a desired apple can feasibly be grasped. This ignores many categories of problems that would have to be addressed in building a real-world system - like hardware design of a mobile base that can autonomously navigate the challenging terrain of a typical orchard, along with the required sensing and navigation capabilities. The problem of identifying and estimate poses for different tree elements is also ignored - we assume perfect knowledge of every branch and apple.

Despite these simplifying assumptions, there are still many problems to explore. This work focuses on the following:

(A) A motion planning approach is developed that plans intentional collisions which deform the environment as needed to reach a goal pose. This approach is shown to be successful in cases when traditional, collision-avoidant motion planning results in failure.

(B) An MPC approach is used to regulate contact forces during trajectory execution. This is shown to reduce contact forces experienced by the arm during trajectory execution compared with simple position control of the trajectories generated in (A).

(C) An approximation of a whole-arm visuotactile sensing skin is developed, to better reflect sensing that might be feasible with real hardware. Depth cameras are placed throughout the arm's interior volume in a configuration that achieves nearly full coverage of the arm's interior surface. Drake models objects in contact as intersecting slightly, this intersection is detected by the depth cameras and converted to a force reading.

II. Related Work

A. Agricultural Robots

A 2022 review paper of harvesting robots from Zhou et al. [1] examines three categories of work: overall system design (e.g. choice of mobile base, quantity and design of manipulators, fruit storage strategy), perception, and fruit detachment method. Only passing mention is given to the problem of motion planning through branches and foliage, with work focusing on optimizing mechanical design of end-effectors to minimize collisions [2]. We do not know any work that specifically explores strategies to deform branches and foliage during motion planning.

B. Manipulating through clutter

The most relevant prior work is that of Jain, Kemp et al. [3], which proposes an MPC approach to regulate contact forces across the surface of an arm equipped with a tactile skin. We will attempt to approximately re-implement this work. An advantage of this approach is the lack of reliance on any detailed model of the environment. Other approaches involve generating computationally expensive models of how the environment will deform [4], [5] and including deformation in subsequent trajectory optimization. These approaches are likely not extensible to agricultural use cases, due to the difficulty in accurately modelling how a complex, heavily occluded object like a tree will deform in response to applied forces.

C. Visuotactile sensing

Visuotactile sensing has reached an impressive level of fidelity and robustness in recent years. Sensors built into grippers can read not just simple contact information - they can also identify grasped objects [7], estimate the poses of those objects [6], and more. Gripper designs take many forms: some solid [8], some made of gels [9], and some filed with air [6]. However, we did not find any works that explored the applications of visuotactile sensing to the full surface of a robot arm. There have been explorations of different sensing technologies applied to larger robot surfaces, like capacitative sensing [10].

III. Simulation Environment

Trees are modelled as randomly generated procedural ”trees” of rigid cylindrical links connected by compliant spherical joints. The tree generator takes many parameters - at each generation, we can specify branching factor and branching angle, branch diameter, length and density, apple frequency and placement, joint stiffness and more. These parameters were chosen non-rigorously to obtain trees that visually resembled real apple trees, and that deformed in a somewhat realistic looking way. A more rigorous approach would be to use empirically obtained values for real variety of apple tree. Our tree simulation does also not have any foliage, the addition of which would substantially increase the realism of the manipulation problem.

Simulating the tree joints presented some challenges. Ideally each joint would be modelled as a spherical/ball joint with 3 degrees of freedom. Drake’s ball joint implementation uses an internal Euler angle representation, which is prone to frequent gimbal lock when used in the tree simulation, which causes Drake to throw an exception. Universal joints were explored, which solved the gimbal lock problem, but these joints do not support stiffness or damping. The eventual solution was to use series revolute joints in close proximity on perpendicular axes, approximating a single ball joint.

For the picking robot, we use a KUKA iiwa 7 mounted with a Schunk WSG gripper. The complete simulated environment is shown below in Figure 1.

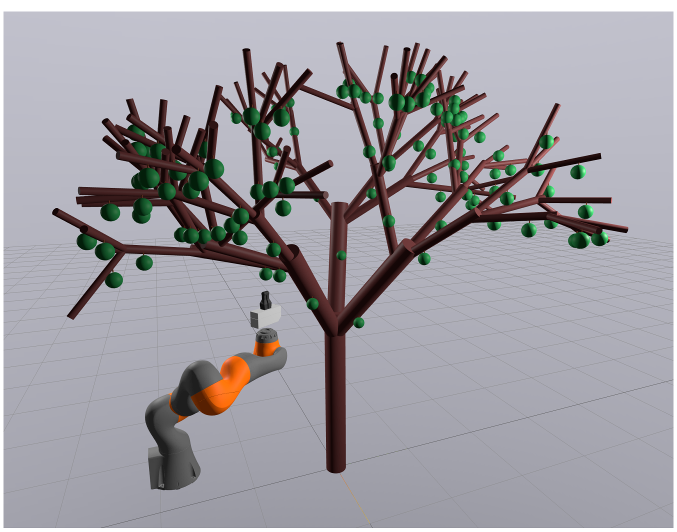


Fig. 1: An illustration of the complete simulation environment, with iiwa arm, WSG gripper, and procedurally generated tree.

IV. Motion Planning

The motion planning system is required to generate a trajectory that positions that end-effector such that it can grasp an apple, without also grasping any surrounding branches. In cases where the apple is easily accessible - i.e. there exists a viable path with no collisions, conventional approaches like collision-aware RRTs or trajectory optimization work well. However in any realistic tree, many apples can be impossible to reach without purposeful deformations of the surrounding branches.

Our approach is to first generate a path using trajectory optimization between the robot’s current generalized position q_{start} and the desired end-effector position p^G, ignoring any potential collisions. Formally, we optimize:

$$\begin{aligned} \min_{\alpha,T} \quad & c_1 T + c_2 \int_0^T \|\dot{q}(t)\| + \\ & c_3 \left\|q_\alpha(T) - q_{home}\right\|^2, \\ \text{subject to} \quad & q_{start} = q_\alpha(0), \\ & \dot{q}_\alpha(0) = \dot{q}_\alpha(T) = 0, \\ & p^G = f_{kin}(q_\alpha(T)), \\ & \forall t, \quad |\dot{q}_\alpha(t)| \le v_{max}, \\ & \forall t, \quad q_{min} \le q_\alpha(t) \le q_{max} \end{aligned} \tag{1}$$

where v_{max}, q_{min} and q_{max} are the arm’s joint velocity and position limits. c_1 is a cost on path duration, c_2 is a cost on path length, and c_3 is a cost on the arm’s final position being far away from a defined comfortable position q_{home}. I is the identity matrix. We use Drake’s `KinematicTrajectoryOptimization` class to implement this optimization.

We then step through this trajectory in a simulation that also contains the tree, and stop whenever the end-effector’s geometry intersects with any tree branch. We generate a safe “waypoint“ relative to the branch and end-effector direction of travel, such that if the end-effector were to make a detour through that waypoint, it would safely deform the intersected branch while then being able to continue along the original trajectory. Formally, if the end-effector’s position and velocity at time t are p_t^E and v_t^E, and v^B is a vector representing the major axis of the intersected branch:

$$\begin{aligned} v^A &= v_t^E \times v^B \\ p_t^A &= d\frac{v^A}{|v^A|} + p_t^E \end{aligned} \tag{2}$$

where v^A represents the direction of adjustment we should make to the trajectory, d controls the magnitude of this adjustment, and p_t^A is the point we want the end-effector to go through as a result. We then modify our original trajectory such that the end-effector goes through each p_t^A generated above, instead of the original p_t^E.

In testing, this approach allowed the end-effector to reach many apples on our simulated trees were inaccessible with conventional motion planning. However, it also introduced its own failure modes. For example, in moving through p_t^A, the end-effector may well collide with a new branch that was not previously considered. In fact, it is entirely possible there is no way of deforming the original branch without first colliding with another branch.

There are many potential refinements to this algorithm that could increase reliability. For example, a more sophisticated framing would be to combine the two planning steps into one. Deformable branches could be accounted for in a single original trajectory optimization step as regions of space that the end-effector can only enter with certain allowable poses and velocities. Closely adjacent branches that cannot be individually deformed could be unified to define the outline of larger region of such space. A cost could be attached to each deformation, such that the optimizer would prefer solutions that deformed fewer branches. Though, these refinements would substantially increase the computational of the optimization, while the method used here is relatively cheap.

V. Control

During trajectory execution, many contacts can occur between the tree and arm. Some are intentional, like those determined by the motion planner, but some are not. Like the secondary collisions outlined previously, or collisions with objects not modelled by our motion planner, like apples. In a real world setting, the magnitude of any contact force experienced by the tree must be kept below sensible thresholds, to avoid damaging the tree or crop.

We implement a form of model predictive control to regulate these contact forces, drawing on the work of Jain et al. [3], though with several differences. An outer control loop running at 20Hz supplements the arm's inner `InverseDynamicsController` controller. At each time step, the outer controller takes in the arm's position q_t and any contact forces and generates a q_{goal} that moves the end-effector incrementally along the desired trajectory while minimizing contact forces.

To model contact forces, we use a similar linear elastic spring model to Jain et al:

$$f_{c_i}(t+1) - f_{c_i}(t) = K_{c_i} J_{c_i}(q_{t+1} - q_t) \tag{3}$$

where $f_{c_i}(t) \in \mathbb{R}^3$ represents contact force i at time t, $K_{c_i} \in \mathbb{R}^{3\times 3}$ is the stiffness of the spring at contact i, and $J_{c_i} \in \mathbb{R}^{3\times m}$ is the Jacobian matrix for the contact i, where m is the dimension of q. Note that each contact is modelled as a point contact, such that torques are not considered under this model.

We then solve the following optimization problem at each time step t.

$$\begin{aligned} \min_{q} \quad & c_1 \sum \|K_{c_i} J_{c_i}(q - q_t)\|^2 + \\ & c_2 \|q - q_t\|, \\ \text{subject to} \quad & p^G_{t+1} = f_{kin}(q), \\ & q_{min} \le q \le q_{max} \end{aligned} \tag{4}$$

where c_1 is a cost on contact force, c_2 is a cost on distance from previous generalized position, and p^G_{t+1} is the desired end-effector position at the next time step.

Note the differences to the approach of Jain et al. Rather than using a linearized model of the robot's kinematics, we use the full true forward kinematics. We linearize only the contact forces. Desired end-effector position is enforced as constraint, rather then being expressed as a cost - for a successful apple grasp, the end-effector must be position with some precision, and so we do not allow deviations from intended position. Jain's approaches to forces is to impose a "don't care" threshold as a constraint - forces must not exceed this threshold, but can take any value below it. We instead use a quadratic cost to minimize any change in contact force experienced by the arm during each incremental movement, as in a real world setting even seemingly gentle contacts can cause harm.

One side-effect our quadratic formulation is that the robot is not encouraged to completely break contacts - which may be a preferable outcome in reality - but instead avoids any change in experienced contact force. Experimentation with different cost functions would be an interesting continuation of this work - for example, $c_1 \sum \|f_{c_i}(t) - K_{c_i} J_{c_i}(q - q_t)\|^2$ could be a better alternative in that it actively drives contact forces to zero. This was briefly experimented with, but the relatively large $f_{c_i}(t)$ term dominated the optimization and caused spurious results. More parameter tuning was likely required. This alternative cost function still does not address the fact that if arm moves away from a contact, the force drops to zero, and so should the corresponding cost. This would require consideration of contact surface geometry.

During trajectory execution, branch deformation often caused the target apple to move substantially. This required a servoing approach - a compensation term was to the desired desired end-effector position that compensated for the updated apple position.

Overall, this approach worked extremely well. The end effector moved along its desired trajectory, while the degrees of freedom provided by the 7 DoF arm and the lack of an orientation constraint were elegantly used by the MPC controller to push the arm away from colliding branches and apples. An example of forces experienced during a movement is shown in Figure 4.

VI. Tactile Sensing

The control approach described in the previous section relies on complete knowledge of all the contact forces experienced on the iiwa's surface, which is somewhat ambitious given the current state of tactile sensing hardware. One possible path to cost-effective whole-arm tactile sensing is suggested by recent innovations in the visuotactile space. Deformable visuotactile sensors could plausibly be applied to the entire exterior surface of a robot, yielding high quality tactile information.

In an attempt to explore the feasibility of this, we implement a very rough approximation of how such a system could work. We place many idealized depth cameras throughout the interior volume of the iiwa. When objects in Drake come into contact, the contact is modelled as an intersection of the two geometries. Our depth cameras can detect this intersection and infer that a contact has occurred.

We place 6 idealized depth cameras, one along each Cartesian axis, within each link of the iiwa and within the gripper, yielding a total of 48 cameras. The cameras are centered in such a way that roughly maximises their view over the interior volume. An example set of depth images from within the iiwa's fourth link is shown in Figure 2.

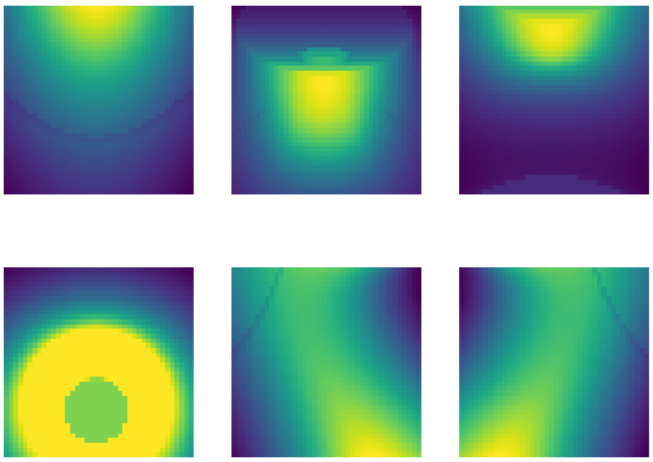


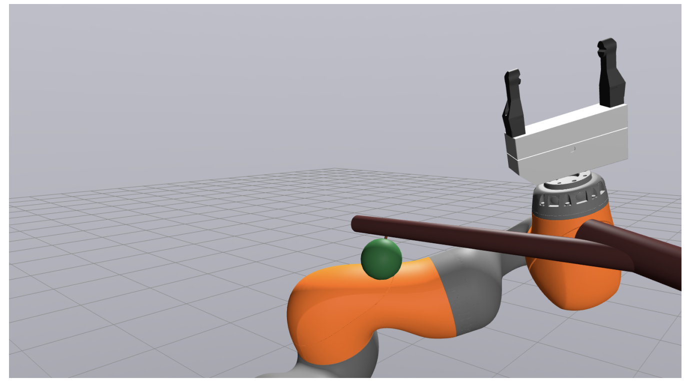
Fig. 2: Example output of depth cameras placed within the fourth link of the iiwa.

We capture reference depth images when the iiwa is not in contact with any other objects. Then, as the iiwa moves, differences against reference images are calculated. Any differences above a threshold are categorized as a contact. kNN is used to match pixels to a particular contact, after which the centroid of each pixel group is obtained. Next, projective geometry is used to obtain the 3D translation of this centroid. An example of the difference in depth image is shown in Figure 3

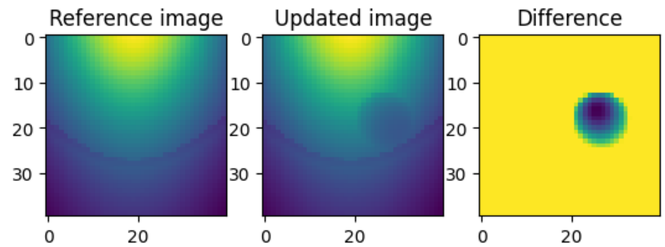
While we could use the exact shape of the intersection to infer the force's orientation, we make a simplifying assumption that the iiwa's surface is smooth, and there are no tangential forces. Thus, we can simply use the surface normal at the region of contact as an approximation of force direction. We can read this from the same depth images using the data matrix eigenvector method.

To ensure this system could reliably detect even low contact forces, a fairly high value had to be applied to on `MultiodyPlant`'s `set_penetration_allowance()` method.

This approach worked very well, albeit within the extremely contrived setting of this problem. A plot of forces encountered by the gripper during a movement is shown in Figure 4. Note that there is almost no difference in force abatement when using our visuotactile approximation vs the true force values read directly from `MultiodyPlant`. This is likely due to a) the idealized depth cameras and b) the lack of surface roughness on iiwa or tree leading to the absence of tangential forces.



(a) Apple in collision with the iiwa



(b) Collision visible in depth image

Fig. 3: Depth cameras used to identify contacts

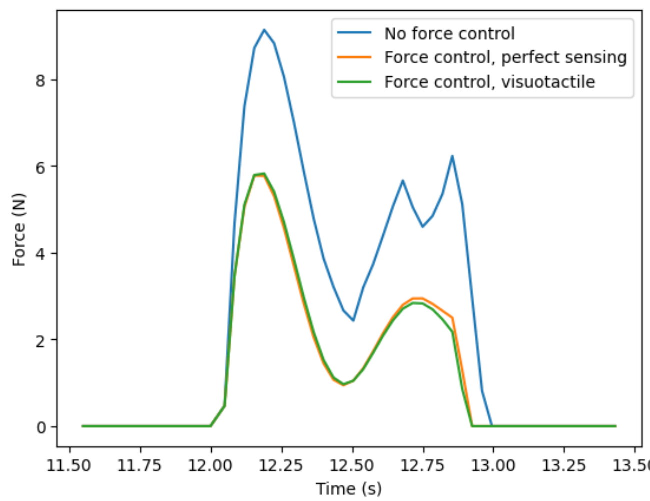


Fig. 4: Forces encountered by gripper during representative movement

VII. Evaluation and Testing

The approach to testing in this project suffered from a lack of rigor due to insufficient time. All parts of the system (motion planning, control and perception) were tested by manually generating different trees and picking interesting looking individual apples as manipulation targets. It was verified that each subsystem worked, and failure modes were explored. However, a more rigorous approach would have been to generate many hundreds of random trees, with thousands of target apples, and generate standard tests with which to compare different methods. For motion planning, the number of times the the target was successfully reached would be a reasonably metric. For force control, the maximum force applied to the tree over thousands of movements could be considered.

VIII. Conclusions and Discussion

This project served as fascinating introduction to the kinds of problems that would arise in attempting to build a system like APPLE-E in reality, and suggests many follow up lines of enquiry. For motion planning, we discussed ways of combining our two pass approach into a single trajectory optimization step, that could result in more robust trajectories. For contact-force MPC, we discussed ways of formulating cost functions that could yield a greater reduction in forces encountered. It would also be interesting to more thoroughly explore the reliance of the MPC approach on high-quality tactile sensing data. For example, if there were in fact tangential forces applied to the robot but not captured in our MPC model, how well would contact forces actually be regulated? Or how would discretization or noise added to the tactile information affect the quality of force regulation? Our simulation could also be made more realistic by more closely modelling real trees - for example, by adding foliage, surface roughness, and more realistic values for density and stiffness.

As expected, those methods that others have found to be effective were effective here too - the contact-force MPC and visuotactile systems working well without much modification. And there are of course many, many further problems that would be to be solved before such a system could be deployed in reality. Like how to actually detach an apple from a tree - with a conventional antipodal gripper, a suction cup, or perhaps a specially designed cutting end-effector. Or the perception problem: foliage may have to be deformed before a target apple can even be seen. Or how to mount an arm on a mobile base that can autonomously navigate through an orchard, while working safely alongside humans. But what fascinating problems these are!

Acknowledgment

Thank you to Russ Tedrake, Thomas Cohn (particularly for the idea to use depth cameras as visuotactile sensors), and all of the 6.4212 course staff and students for a hugely enjoyable semester. Thanks also to the Drake team for their lightning quick turnaround time when fixing bugs.

References

[1] Hongyu Zhou et al. "Intelligent robots for fruit harvesting: recent developments and future challenges". In: Precision Agriculture 23.5 (Oct. 2022), pp. 1856–1907. issn: 1573-1618.

[2] Satoshi Yamamoto et al. "Development of a Stationary Robotic Strawberry Harvester with a Picking Mechanism that Approaches the Target Fruit from Below". In: Japan Agricultural Research Quarterly: JARQ 48.3 (2014), pp. 261–269

[3] Advait Jain et al. "Reaching in clutter with whole-arm tactile sensing". In: The International Journal of Robotics Research 32.4 (2013), pp. 458–482

[4] Barbara Frank et al. "Using Gaussian Process Regression for Efficient Motion Planning in Environments with Deformable Objects". In: Proceedings of the 9th AAAI Conference on Automated Action Planning for Autonomous Mobile Robots. AAAIWS'11-09. AAAI Press, 2011, pp. 2–7.

[5] Sachin Patil, Jur van den Berg, and Ron Alterovitz. "Motion Planning Under Uncertainty In Highly Deformable Environments". In: June 2011.

[6] Kuppuswamy, N., Alspach, A., Uttamchandani, A., Creasey, S., Ikeda, T., & Tedrake, R. (2020). Soft-Bubble grippers for robust and perceptive manipulation.

[7] H. Liu, Y. Wu, F. Sun, and G. Di, "Recent progress on tactile object recognition," International Journal of Advanced Robotic Systems, vol. 14, pp. 1–12, 07 2017

[8] J. Hughes, U. Culha, F. Giardina, F. Guenther, A. Rosendo, and F. Iida, "Soft manipulators and grippers: A review," Frontiers in Robotics and AI, vol. 3, p. 69, 2016. [Online]. Available: https://www.frontiersin.org/article/10.3389/frobt.2016.00069

[9] W. Yuan, S. Dong, and E. H. Adelson, "GelSight: High-resolution robot tactile sensors for estimating geometry and force," Sensors, vol. 17, no. 12, p. 2762, 2017

[10] Ulmen J, Edsinger A, Cutkosky M (2012) A highly sensitive, manufacturable, low-cost tactile sensor for responsive robots. In: In Submission, IEEE International Conference on Robotics and Automation.