**Human Modeling for Efficient Predictive Collision Detection**

**Name of Author and Name of Co Author**

*College of Affiliation, University of affiliation*

Name of Faculty Mentor, Department

Abstract

As demands on manufacturing rapidly evolve, flexible manufacturing is becoming more essential for acquiring the necessary productivity to remain competitive. An innovative approach to flexible manufacturing is the introduction of fenceless robotic manufacturing cells to acquire and leverage greater human-robot collaboration (HRC). This involves operations in which a human and a robot share a space, complete tasks together, and interact with each other. Such operations, however, pose serious safety concerns. Before HRC can become a viable possibility, robots must be capable of safely operating within and responding to events in dynamic environments. Furthermore, the robot must be able to do this quickly during online operation. This paper outlines an algorithm for predictive collision detection. This algorithm gives the robot the ability to look ahead at its trajectory, and the trajectories of other bodies in its environment and predict potential collisions. The algorithm approximates a continuous swept volume of any articulated body along its trajectory by taking only a few time sequential samples of the predicted orientations of the body and creating surfaces that patch the orientations together with Coons patches. Run time data collected on this algorithm suggest that the algorithm can accurately predict future collisions in under 30 ms.

*Keywords:* Predictive Collision Detection, Swept Volume Interference, Coons Patches

Introduction

Automation of manufacturing processes over the past few decades has yielded enormous benefits in efficiency and quality. This was accomplished by increasing speed and precision, and improving material handling devices and equipment along with sensory capability. Products now are experiencing quicker life cycles and increased customization. This means that manufacturing processes must be more flexible and adaptable to meet rapidly changing demands. This can be accomplished by developing human-robot collaboration (HRC) methodologies; taking advantage of the speed, power, and precision of robots as well as the creativity and adaptability of humans (Kruger et al., 2009). To do this, robots must be flexible enough to operate safely around humans actively working within fenceless environments. This further elevates challenges as safety concerns surround unrestricted operation of robots near humans. Before HRC can be implemented, robots must learn to predict and avoid collisions with humans.

In previous work, an algorithm for predictive collision detection was developed to enable a robot to forecast its motion amongst moving ellipse shaped object profiles to identify collisions. Coons patches were leveraged to approximate the robot’s position throughout time as a swept volume (Streitmatter and Wiens, 2020). In this paper, the algorithm is extended to model the articulated, multi-link body trajectories of the human to predict collisions between a robot and human. This paper contributes to a multi-university/industry/national laboratory collaborative research effort to develop HRC techniques for sensing of dynamic environments, forecasting change in the environment, and enabling robots to respond safely and productively (Yin et al., 2018). The algorithm in this paper will contribute to an HRC Proactive Adaptive Collaborative Intelligence (PACI) module used to control robot motion and operation (Nicora et al., 2020).

Related Work

One of the simplest and fastest ways of performing predictive collision detection is to employ “*Multiple Interference Detection”.* In this process a representative set of configurations throughout the robot’s trajectory are checked for interference. If, however, the sampling frequency is too low, collisions can be missed. Another more robust approach is “*Swept Volume Interference*”, in which an object is swept along its trajectory to create a composite shape representing the total volume affected by the object. A collision can be identified by overlaying swept volumes and identifying where spatial and temporal data intersect (Jiménez et al., 2001). This is computationally expensive and difficult to implement in real time. For online operation of a robot in dynamic environments in which human safety is in question, fast and efficient algorithms and techniques are needed.

One combination of both approaches is employed in (Mainprice & Berenson, 2013), where a pre-existing 3D surface is sampled as it moves along its trajectory at a high frequency such that an approximate swept volume is created. While this approach is faster than computing a swept volume analytically, it requires the pre-existence of a 3D surface, and a large number of samples to construct the surface.

The algorithm presented in this paper addresses the need for a computationally faster collision detection algorithm that can operate on much more basic predictions of the human’s location and orientation throughout time. By implementing Coons patches to interpolate the human’s position throughout time, fewer samples are required to construct a representative swept surface, greatly improving the computational efficiency.

Methods

The developed algorithm assumes as its only input the predicted motion derived from simulated sensor data describing joint angles of a human’s articulated multi-link body at various instants in time. From this input, forward kinematics techniques are used to define boundary points outlining the human at each predicted instant in time. These boundaries are then used to create a boundary surface that represents the swept volume of the human. This approach is able to strike a tradeoff between “*Swept Surface Interference”* and “*Multiple Detection Interference”* methods by interpolating between a lower frequency sampling to create a continuous surface. Furthermore, since the swept volume is described only by the boundary surface, points within the surface do not need to be characterized, greatly reducing computer memory and computational costs. For collision detection, only the surface needs to be modeled as it can be assumed that if another volume passes into the space of the swept volume, it must first go through and collide with the swept boundary. The same approach is then applied to the robot. Simulations of the robot and human in a shared workspace are used to evaluate effectiveness and computational efficiency of the algorithm.

Overview of the Approach

A preliminary grid size must be set to determine coarseness of the grid on which the human and robot swept volumes are to be compared. A fine grid size will result in an increase in computational time but will yield more accurate representation of the swept volume. After each swept volume is created, all points defining the swept volume are relocated to the nearest grid location. This grid allows for direct comparison between multiple swept volumes. The remainder of the algorithm executes three general steps.

First as shown in Figure 1, boundary points (joint offsets, shown in red) are created around each of the joints. These locations are determined directly from the input joint angles through forward kinematics and are positioned such that they create a boundary around the human in a plane facing the direction normal to the initial orientation of the human. As seen in Figure 1, the normal direction would be out of the page. The initial orientation is a consequence of how the coordinate systems used to model each joint are defined. To account for motion in the orthogonal direction, boundary points are also defined in a plane orthogonal to the initial orientation of the human, creating a side profile of the human (Figure 2). This defines a 3-D boundary of the human at one instance in time, such that as each link of the human moves, it will have a section of thickness orthogonal to the direction of motion. This process is repeated for each instance in time for which data on the human is given. To account for uncertainty in future predictions and safety considerations, safety factors are built into the dimensions of the boundary curves such that the boundary curves grow with time. The growth is a function of the body part of the human. For example, since a collision with the head is of greater safety concern than the arm, the dimensions of the head will expand more rapidly than those of the arm.

Next, discrete Coons patches (Farin & Hansford, 1999), are used to define surfaces between each orientation throughout time. The patches provide an efficient expression for generating point clouds that fill the gaps between the given predicted positions of the human. These patches are key to maximizing computational efficiency by eliminating the need for completing forward kinematics at a high sampling frequency.

As the human moves, the normal and orthogonal curves will sweep out. As can be seen from Figures 1 and 2, this will still leave portions of the human in the initial and final poses unmodeled. To address this and to create a closed surface therefore completing the volume definition, initial and final conditions of the human are implemented to generate surfaces at the beginning and end of the sweep. This step wraps a surface around the overall swept volume of the human. The resulting swept volume can finally be compared with other swept volumes to identify spatiotemporal intersection of the volumes.

Definition of the Boundary Curves

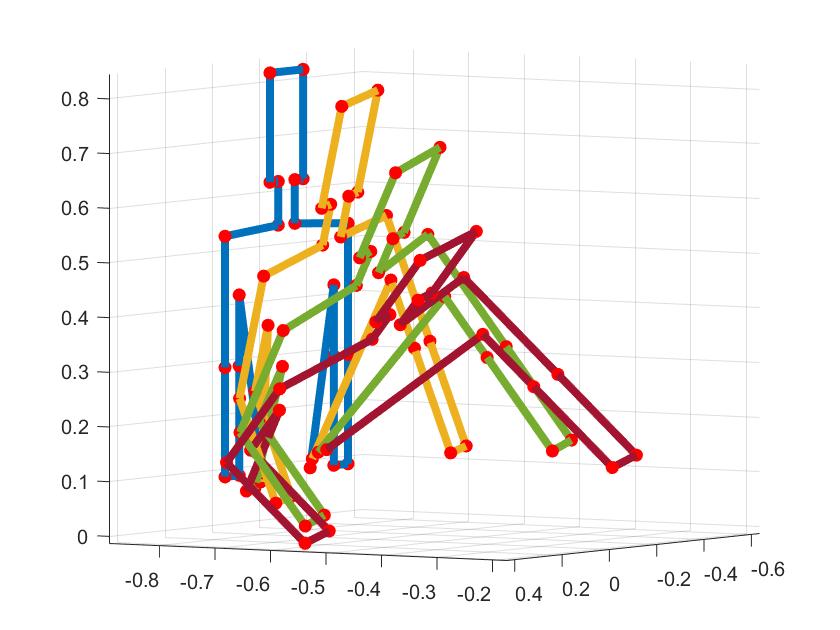
The simulated testing utilized in this paper and in the overall project at large is completed with a multi-axis collaborative tabletop robot arm. Because such robots are largely used for small, tabletop assembly and inspection applications, the human model selected to interface with such a robot is a seated worker. Thus, only the waist up of the human was modeled. The motion of the model was defined by a torso joint with three axes of rotation, two shoulder joints with three axes of rotation each, two elbow joints with one axis of rotation each, and a neck joint with three axes of rotation. Coordinate systems are established with their origin locations fixed at the center of each joint. The location of these systems with respect to each other and with respect to an overall fixed system are described with forward kinematics. Specifically, transformation matrices are used to describe relative distances between two systems, and rotation matrices are used to rotate the systems about their joints. These matrices are defined in Denavit Hartenberg notation.

To locate the position in the fixed coordinate system of any joint after arbitrary rotations about each joint, a series of transformation and rotation matrices are used to travel from the fixed system through each transformation and rotation of each successive system up to the joint of interest. Consider a point,, with a known position in the coordinate system *n,* . To locate in the coordinate system of the previous joint, *n-1,* the point must be pre-multiplied by a translation matrix, *,* that describes points in system *n* as seen from system *n-1* A rotation about the joint in either system will also affect the location of the point. To account for rotations, and a rotation matrix rotating about joint *n*, , must also be pre-multiplied. A generalized formula for translating a point in the *n* system through an arbitrary number of translations and rotations is presented in (1) where is the translation from system one to the fixed system and is the location of point as seen from the fixed system.

|  |  |  |
| --- | --- | --- |
|  |  | (1) |

Using this approach, the supplied joint angles for each of the human’s joints can be used to find the location of each joint in cartesian space. Additionally, to provide physical dimension to the human’s body, boundary points about each of the human’s joints can be defined in the coordinate system of the joint they correspond to and translated to the fixed frame. To build in a factor of safety and account for uncertainty in the predicted location of the human over time, the distance between these boundary points and the joint they correspond to is increased by the product of a scaling factor and the amount of time between the prediction and the execution time of the algorithm. As rotations occur about each joint, these boundary points, and the subsequent points in the kinematic chain also rotate. The left side of Figure 1 depicts the boundary points (*marked in red*) connected with bluelines. Repeating this process for each step-in time, boundary points are defined for the entire trajectory, as seen in the right side of Figure 1, where each instance in time is represented by a different color outline.

A picture containing table, white, black, sitting

Description automatically generated

**Figure 1.** Location of the joint offsets (red) determined with forward kinematics. Initial orientation is assumed to be facing the origin of the fixed coordinate system (base frame of robot) with the human’s arms down.

To account for the component of motion in the orthogonal direction, a similar boundary is drawn to model the side profile of the human’s arm (drawn in green) and torso and head (both drain in blue) (Figure 2 Left). The interface between the normal and orthogonal boundary curves is also shown (Figure 2 Right). In the same approach applies above, the boundary points for this profile are defined at each time step.

Arm (Green)

Joint offsets

(in red)

**Side Profile**

Head

Shoulder

Torso

A picture containing table, light, sitting, filled

Description automatically generatedA close up of a map

Description automatically generated

**Figure 2.** Joint offsets (red) in the orthogonal direction to account for lateral motion. The right side of the figure shows the combination of the normal and orthogonal curves.

Application of Coons Patches to Connect the Boundary Curves

With boundary curves defined, the next step is to generate surfaces between them. This operation is completed with discrete Coons patches. An individual patch is created for each straight segment of the human as the segment travels between two different orientations. The boundary curves defined previously must now be put into a format that the discrete Coons patch formulation applies to. Example boundary curves for a Coons patch are shown in Figure 3. Four boundary curves are used to define each patch (Figure 3 left). The first boundary curve (*Curve 1*) follows a line between a point on the human at an instant in time and the same point at the next (second) instance in time. The second boundary curve (*Curve 2*) follows a line between the first point on the human at the second instance in time and a second point on the human at the second instance in time. The third boundary curve (*Curve 3*) follows a line between the second point at the second instance in time and the same point on the human at the first instance in time. The fourth boundary curve (*Curve 4*) follows a line between the second point at the first instant in time and the first point at the same instance in time.

A close up of a logo

Description automatically generatedA close up of a device

Description automatically generated

**Figure 3.** Definition of boundary curves and generation of a patch

With the corner points specified, the curves can be put into the proper mathematical form for application of Coons patches. The mathematical form is shown in Eq. (2), where to represents discrete values along Curve 1; to represents Curve 2; to represents Curve 3; and to represents Curve 4. The variables *v* and *u* define the horizontal and vertical components of the mesh of the Coons patch. Higher values of *v* and *u* result in more densely packed Coons patches. Finer mesh sizes will lead a more densely defined surface, and better collision detection, but will increase computational time.

|  |  |  |
| --- | --- | --- |
|  |  | (2) |

Through careful selection of *v* and *u*, computational effort can be drastically reduced without sacrificing any accuracy. This is done by selecting a minimum *v* and *u* such that the maximum distance between any two points in the Coons patch is smaller than the grid size selected at the start of the algorithm. Denser Coons patches will be mapped to the spatial grid size, so high-density patches will eventually be generalized by patches that fit the grid size in the final comparison of the swept volumes.

Since it is advantageous to uniquely determine *v* and *u* for each patch, once the boundary curves have been evaluated, a function looks at their total lengths. The *u* is determined by evaluating the lengths of Curves 1 and 3. The *u* is set to equal the length of the longer curve divided by half the grid size. Similarly, *v* is set by evaluating the lengths of Curves 2 and 4 and dividing the larger of the two by half the grid size. In this way, it is ensured that the Coons patch can be accurately mapped to the grid size without much change in location of the points within the patch and with no risk of marking as unoccupied grid locations that should be occupied.

To define the patch, let *i* and *j* represent intermediate values between the boundary curves. The points that lie at each set of indices *i* and *j* in , where *i* and *j* range from 1 to *v-1* and 1 to *u*-*1* respectively, are calculated in the discrete equation of a Coons patch, Eq. (3). Each patch contains a discretized grid of points. Each point represents the weighted average of the closest points on the boundary curves with respect to the distance between the point and each curve.

|  |  |
| --- | --- |
|  | (3) |

Completing the above calculation for the patch, a ) set of points within the boundary curves is created. This process is completed for each dimension of interest. A patch is made for the X, Y, and Z dimensions as well as for the time dimension. The boundary curves for the time dimension patch are defined as follows. Curves 2 and 4 lie between two points known at specific times. They are characterized by these times. Curves 1 and 3 are defined as linear interpolations between the time at the first configuration and the time at the second. An example patch with the time displayed as color is shown in right side of Figure 3. Applying this process iteratively to each segment of the human’s motion between each known pose defines the volume swept out by the human. In Figure 4, the positional X, Y, and Z patches are used to plot the human’s swept volume, and the time patch, displayed as color, further characterizes each point.

A close up of a mans face

Description automatically generated

Head/Neck

Left Arm

Right Arm

Left Forearm

Right Forearm

Torso

**Figure 4.** Iterative application of Coons patches to define the boundary surface. Time is shown as color. Darker colors represent the initial state and lighter colors represent the final state.

Setting Initial and Final Conditions

Up to this point, the starting pose and ending pose of the human’s motion is unbounded by a surface. To model them, initial and final conditions are applied. The same formulation of Coons patches is employed but rather than patch together two different poses, patches are generated to connect all components of one pose. This is done for the first and the last pose then added to the composite swept volume (Figure 5).

**Back View**

**Front View**

*A close up of a logo

Description automatically generated A close up of a logo

Description automatically generated*

Head/Neck

Left Arm

Right Arm

Left Forearm

Right Forearm

Torso

**Figure 5.** Initial (Left) and final (Right) conditions implemented to close boundary surface

Using this same methodology, a robot was also modeled. Figure 6 (*left*) shows the robot in its initial and final orientations, and figure 6 (*right*) shows the swept volume. This swept volume is defined in the same fixed system as the human’s swept volume.

A picture containing photo, man, snow, woman

Description automatically generatedA picture containing hat

Description automatically generated

**Figure 6.** Model of the robot’s swept volume. The start and end orientations of the robot are shown, but a total of six poses of the robot were used to generate the volume.

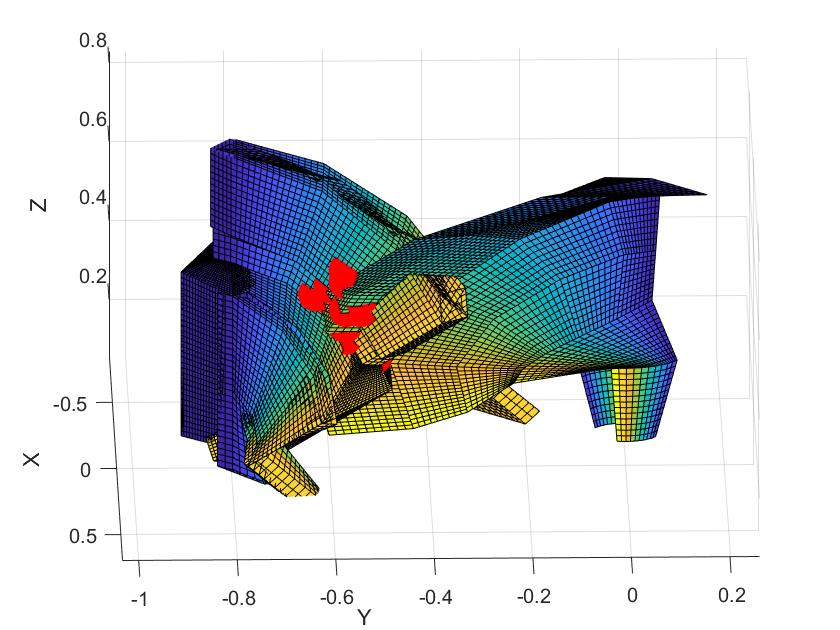
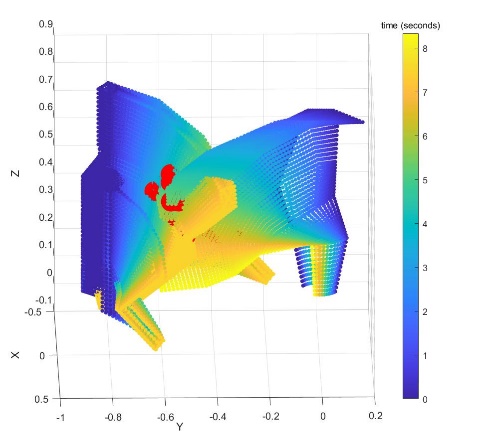
To check the swept volumes for collision, they are mapped into the previously mentioned standardized grid and a standard discretized time array. Each point is fit to the nearest grid space, and the time associated with each point is fit to the nearest discretized time. Collisions occur wherever both sweeps contain the same positional and time data. An example collision is shown in Figure 7, where collision points are indicated in red.

Robot

Robot

Human

Human



**Figure 7.** Collision detected between the robot and human swept volumes

Evaluation

Tests performed to evaluate the algorithm were implemented on an Intel® Core™ i7-7500 CPU processor. To evaluate computational efficiency, timed tests were run at grid sizes between 0.01 m to 0.15 m. At each grid size the algorithm was run 100 times to evaluate 100 randomly generated human and robot trajectories for collision. The computational times at each grid size were averaged and recorded (Figure 8). An exponential increase in computation can be seen as grid sizes decrease below 0.02 m. For grid sizes greater than 0.07 m, the algorithm runs in real time (less than 30 ms).

**Figure 8.** Computational time associated with each grid size. The trend displays monotonic behavior but exponentially for grids with a spacing of less than 2 cm.

The effectiveness of the algorithm at various grid sizes was evaluated for 100 test cases with randomly generated trajectories for the human and robot. Each test case was evaluated at fifteen grid sizes equally spaced between 0.01 m and 0.15 m. Given that the modeled human, from waist up has a height of 0.73 m, these grid sizes constitute a range of roughly 1 % and 20 % of the total height. The goal of each test was to identify whether there was a collision or not. Of these 100 scenarios, 87 tests maintained the same prediction for all grid sizes. For 8 tests, lower resolution iterations misidentified a collision, but converged to a consistent solution for higher resolution grid sizes. The highest resolution at which the algorithm misidentified a collision was at 0.060 m, while the average grid size at which the algorithm converged was 0.110 m. There were four cases in which the algorithm did not to converge to a solution. In these 4 cases, as well as the previously discussed 8 cases, both swept volumes came indistinguishably close to each other, and collisions could not even be determined upon visual inspection.

These results indicate that the algorithm is successful in identifying collisions for grid sizes less than 0.06 m. The algorithm can be expected to perform accurate collision detection for grid sizes of 0.100 m and smaller in less than 30 ms.

Conclusion

The contribution of this paper is a quick, effective predictive collision detection algorithm designed to run in real time as a background operation of a robot controller. The novelty of the approach was to use Coons patches to interpolate between sampled orientations of a body throughout time. In this way, continuous collision detection is implemented with minimal computation.

Future works will include optimizing the grid size according to the input trajectory to accurately model necessary features with the largest grid size possible and implementing the algorithm with sensor data recording interaction between a physical human and robot to evaluate the algorithm with hardware in the loop testing.

**Acknowledgements**

Funding was provided by University of Florida’s Scholars Program. Research conducted in support of NSF/NRI: INT: COLLAB: Manufacturing USA: Intelligent Human-Robot Collaboration for Smart Factory (Award I.D. #:1830383). Any opinions, findings and conclusions or recommendations expressed are those of the researchers and do not necessarily reflect the views of the National Science Foundation.

References

Kruger, J., Lien, T. K., and Verl, A. (2009). Cooperation of Human and Machines in Assembly Lines. *CIRP Journal of Manufacturing and Technology, 58(2), 628-646.*

Streitmatter, G., and Wiens, G. (2020). Human-Robot Collaboration: A Predictive Collision Detection Approach for Operation within Dynamic Environments. *Manuscript submitted for publication.*

Yin, Z., Leu, M., Wiens, G., and Gao, R. (2018). “NRI: INT: COLLAB: Manufacturing USA: Intelligent Human-Robot Collaboration for Smart Factory”, *2018 NSF/National Robotics Initiative (NRI) Principal Investigators’ Meeting*, Arlington, VA, Oct. 29-30. poster (invited).

Nicora, M.L., Ambrosetti, R., Wiens, G.J., and Fassi, I. (2020). Human-Robot Collaboration in Smart Manufacturing: Robot Reactive Behavior Intelligence. *ASME Manufacturing Science and Engineering Conference*, MSEC2020-10274.

Jiménez, P., Thomas, F., and Torras, C. (2001). 3D Collision Detection: A Survey. *Computers & Graphics*, *25(2), 1-7.*

Mainprice, J., and Berenson, D., (2013) Human Robot Collaboration Manipulation Planning Using Early Prediction of Human Motion. [*International Conference on Intelligent Robots and Systems*](https://ieeexplore.ieee.org/xpl/conhome/6679723/proceeding), Tokyo, Japan, Nov 3-7.

Farin, G. and Hansford, D. (1999). Discrete Coons Patches. *Comput. Aided. Gerom. Design,* *16, 691-700.*