**Human Modeling for Efficient Predictive Collision Detection**

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

*College Affiliation, University Affiliation,*

Name of Faculty Mentor, Department

Abstract

As demands on manufacturing rapidly evolve, flexible manufacturing is becoming more necessary. An innovative approach to flexible manufacturing is 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 in and responding to 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 samples of the 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 the speed and precision with which a task was completed and by unlocking new capabilities through introduction of powerful equipment. As demands on manufacturing continue to evolve, however, more is required. Products demanded in most facets of consumer goods and even industrial requirements are experiencing quicker life cycles and increased customization. This means that manufacturing processes must, in addition to maintaining efficiency and quality, be more flexible and adaptable to meet rapidly changing demands. This can be accomplished by developing human robot collaboration (HRC) methodologies, and taking advantage of the speed, power, and precision of robots, as well as the creativity, and adaptability of humans (Kruger et al., 2009). In order to do this, it is necessary that robots themselves be flexible enough to operate safely around dynamic humans. 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 to identify collisions. Coons patches were leveraged to approximate the robot’s position throughout time as a swept volume (Streitmatter & Wiens, 2020). In this paper, the algorithm is improved and used to model a human trajectory to predict collisions between a robot and human. The algorithm developed provides a computationally efficient swept volume approximation of a human for predictive collision detection. This paper contributes to a collaborative effort to develop HRC techniques. The collaborative effort involves approaches to sensing of dynamic environments, forecasting how the environment is evolving, and enabling robots to respond safely and productively (Xiong et al., 2020). The work in this paper will feed into a Proactive Adaptive Collaborative Intelligence (PACI) module that will be 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 such swept volumes and locating any locations where spatial and temporal data intersect (Jiménez et al., 2001). This solution is computationally expensive and thus 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.

Attempts at approaching the computational efficiency of “*Multiple Interference Detection”* while maintaining the robustness of “*Swept Volume Interference”* generally involve creating point clouds with spatial and temporal information that are dense enough that an object cannot pass between points without enveloping a point. One variation of this approach is employed in (Mainprice & Berenson, 2013), in which 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, it requires the pre-existence of a 3D surface, and still requires a large number of samples to construct the surface.

The algorithm presented in this paper addresses the need for a higher speed collision detection algorithm that can operate on much more basic predictions of the human’s location and orientation throughout time. The only assumed input data is the rotation angles of a few representative joints on the human, such as the shoulder and elbow joints, the neck, and the torso. Furthermore, 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 simulated sensor data describing joint angles on a human at various instants in time. From this input, forward kinematics techniques are used to define boundary points outlining the human at each 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. The same approach is then applied to a robot. Simulations of both the robot and human in the same environment 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 a 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 step allows for direct comparison between multiple swept volumes. To accurately model the swept volume of the human and check for collision with other swept volumes, the algorithm completes three general steps.

First, boundary points 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. To account for motion in the orthogonal direction, boundary points are also defined in a plane orthogonal to the initial orientation of the human. This defines a three-dimensional boundary of the human at one instance in time. 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 the head is of more 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, which are described in more detail in subsequent sections, provide an efficient procedure for generating point clouds that fill the gaps between the known 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.

Finally, 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

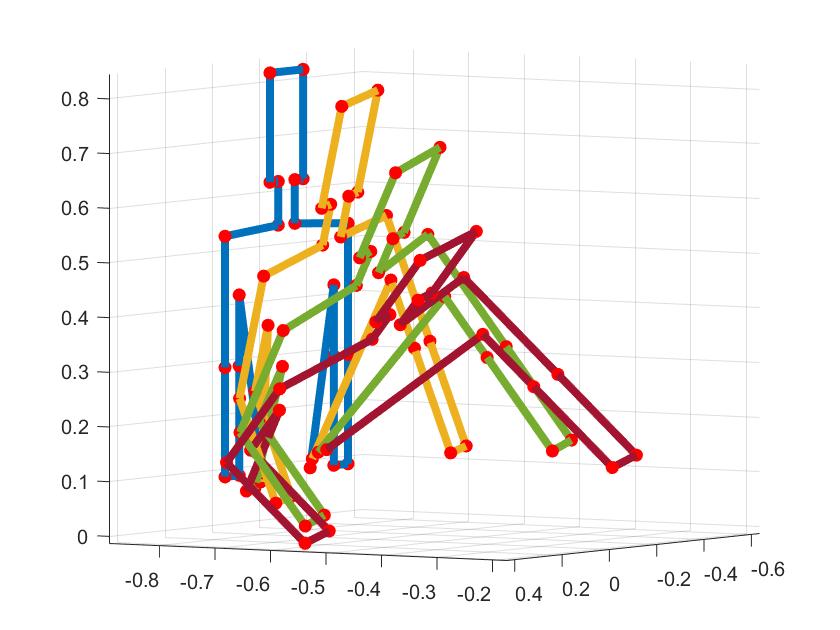
The simulated testing utilized in this paper and in the overall project at large is completed with a multi-axis collaborative tabletop robot arm. Because such robots are largely used for small, tabletop assembly and inspection applications, the human model selected to interface with such a robot was 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 axis of rotation, two shoulder joints with three axis 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, 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 adding to the original distance 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 joint locations, marked in red, and the boundary points, marked in blue, connected with orange lines. 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.

A close up of a logo

Description automatically generated

**Figure 1.** Location of the joints and joint offsets determined with forward kinematics

To account for component of motion in the orthogonal direction, a similar boundary is drawn to model the side profile of the human (Figure 2 Left). In the same approach applied above, the boundary points for this profile are defined at each step in time (Figure 2 Right).

A picture containing table, light, sitting, filled

Description automatically generatedA close up of a map

Description automatically generated

**Figure 2.** Joint offsets in the orthogonal direction to account for lateral motion

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. The composite surface is created by piecing together a number of such 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 follows a line between a point on the human at an instant in time and the same point at the next instance in time. The second boundary curve follows a line between the first point at the second instance in time and a second point on the human at the second instance in time. The third boundary curve 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 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 piece of paper

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 configuration for application of Coons patches. The configuration is shown in (2), where to represents discrete values along curve one, to represents curve two, to represents curve three, and to represents curve four (2). 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 grid size anyways, 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. *u* is determined by evaluating the lengths of curves one and three. *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 two and four 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 (3). Each patch contains a discretized grid of points. Each point represent 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, an 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 two and four lie between two points known at specific times. They are characterized by these times. Curves one and three 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 Figure 3 on the right. Applying this process iteratively to each segment of the human 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

**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

The previous step defines a surface that traces the outline of the human as motion occurs. This, however, leaves the starting point and ending point if the human’s motion unbounded by a surface. To close these ends of the surface, 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. With this step, the swept volume is completed (Figure 5).

.

*A close up of a logo

Description automatically generatedA close up of a logo

Description automatically generated*

**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 end 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.

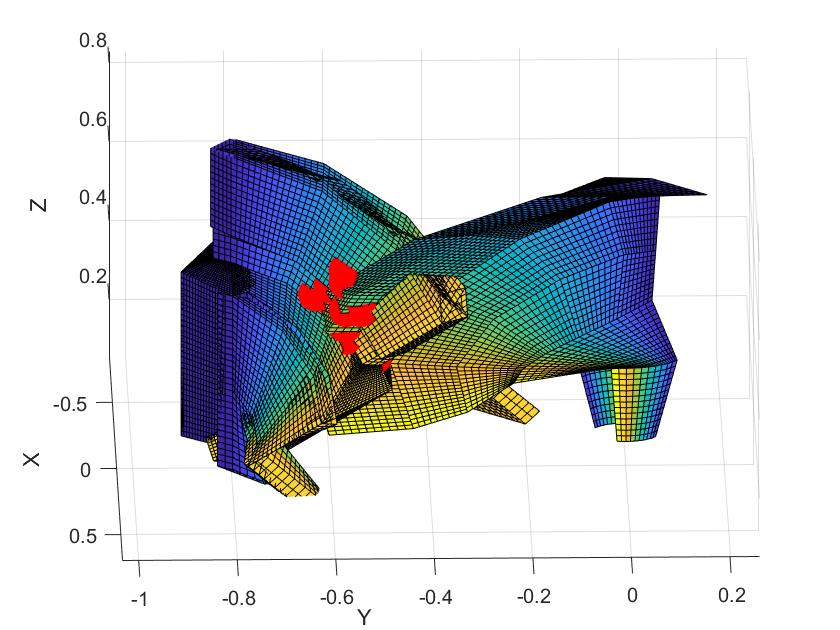
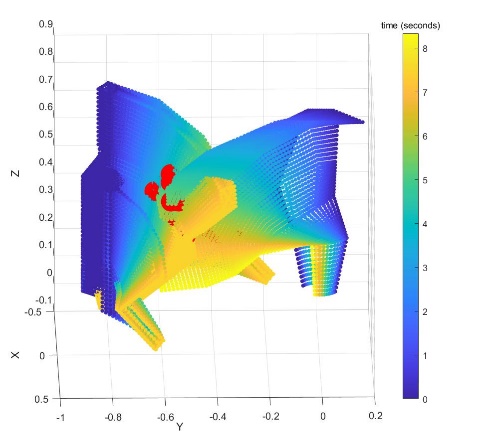
A picture containing photo, man, snow, woman

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.

Finally, to check the two swept volumes for collision, the points defining each volume are mapped into a standardized grid. The grid size initially specified is used. A standard discretized time array is also used. 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 at any location where both sweeps contain the same positional and time data. An example collision is shown in Figure 7, where collision points are indicated in red.



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

Evaluation

To evaluate the algorithm, a tester was built to assign joint angle trajectories to both the human and the robot. These tests were implemented on an Intel® Core™ i7-7500 CPU processor. Trajectories were randomly generated for each test, limited only by the limits of each joint. In previous work, the effects of varying the sampling frequency used to create the boundaries for the Coons patches was evaluated. For this study, due to the adaptations made, the parameter that had the most influence over computational time was found to be the grid size. Thus, to evaluate computational efficiency, tests were run at varying grid sizes and the run time for the algorithm was recorded. Grid sizes between 0.01 m to 0.15 m were each tested. At each grid size the algorithm was run 100 times with random human and robot trajectories each time. The results were then averaged. Figure 8 displays a graph of the time versus grid size results.

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

To evaluate the effectiveness of the algorithm, 100 test cases were generated with random joint trajectories for both the human and the robot. These test cases were each evaluated at fifteen grid sizes equally spaced between 0.01m and 0.15 m to understand the limits on the coarseness of the grid. The goal of each test was to identify whether there was a collision or not. Of the 100 scenarios, tested 87 tests maintained the same prediction for all grid sizes. For these tests, the coarsest grid sizes proved reliable. For 8 tests, lower resolution iterations misidentified a collision, but converged to a final solution and maintained that solution for the remainder of the higher resolution grid size iterations. For these tests, 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 to a solution was 0.110 m. Finally, there were four cases in which the algorithm did not to converge to a solution. Upon visual examination of these 4, as well as the previously discussed 8 cases, it was noted that in all of these cases, both swept volumes came nearly indistinguishably close to each other. For the 4 nonconverging cases as well as most of the 8 converging cases, collisions could not even be identified upon visual inspection.

The results of the effectiveness evaluation indicate that the algorithm is successful in identifying collisions for grid sizes less than 0.06 m. This is, however, a worst-case scenario bound, and the algorithm can be expected to perform accurate collision detection for grid sizes of 0.100 m and smaller. According to the time study, this corresponds to a worse case computational time of 45.3 ms, and an expected computational time of 27.2 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. This is a building block to a larger effort to develop HRC techniques.

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 the NSF/NRI: INT: COLLAB: Manufacturing USA: Intelligent Human-Robot Collaboration for Smart Factory (Award I.D. #:1830383). Additional funding was provided by University of Florida’s Scholars Program. 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., & Verl, A. (2009). Cooperation of Human and Machines in Assembly Lines. *CIRP Journal of Manufacturing and Technology, 58(2), 628-646.*

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

Xiong, Q., Zhang, J., Wang, P., & X. Gao, R. (2020). Transferable Two-stream Convolutional Neural Network in Human-robot Collaboration. *48th SME* *North American Manufacturing Research Conference*, Cincinnati, OH, June 22-26.

Nicora, M.L., Ambrosetti, R., Wiens, G.J., & Fassi, I. (2020). Human-Robot Collaboration in Smart Manufacturing: Robot Reactive Behavior Intelligence. *ASME Manufacturing Science and Engineering Conference*, Cincinnati, OH, June 22-26.

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

Mainprice, J., & 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. & Hansford, D. (1999). Discrete Coons Patches. *Comput. Aided. Gerom. Design,* *16, 691-700.*