Continuous Collision Detection Algorithm in CollDet

Anjishnu Mukherjee

August 7, 2021

Question Given 2 rigid objects moving independently, determine an accurate time of collision between them.

Answer For the simplest scenario, we consider 2 spheres, one of which is always stationary. The other sphere can be either *rotated* or *translated* in any arbitrary direction. Consider $\texttt{t}_0$ to be an instant where they are not colliding and $\texttt{t}_1$ to be the next instant where they are colliding, where $\texttt{t}_1 = \texttt{t}_0 + 1$. The exact time of collision $\texttt{t}_{\texttt{coll}}$ will lie somewhere inbetween $\texttt{t}_0$ and $\texttt{t}_1$, i.e $t_0 < t_{coll} < t_1$. $m12$ is a 4x4 matrix which contains the translation and rotation vectors of the moving object. A collision will not be detected when $m12$ becomes zero. $m12_{t_\text{i}}$ is the matrix at the time t_i. We only have the matrix available for t_0 and t_1, so we need to extrapolate for the fractional instants between those 2 instants. For example, we will halve the values in the matrix, to find an extrapolation for time $\text{t}_{0.5}$ and so on.

The algorithm will start at t_1 where we know collision has already occured. Then it will calculate the state at $\text{t}_{0.5}$ by extrapolating. If collision occurs at $\text{t}_{0.5}$, we only need to search between t_0 and $\text{t}_{0.5}$, because at any time between 0.5 and 1, collision is now known to occur. So, effectively the search space for the time is now halved and we can treat $\text{t}_{0.5}$ as our *effective* t_1 now and repeat the same process as long as collision occurs at the *effective* $t_{0.5}$.

Figure 1: The two spheres just about to collide at $\text{t}_0 = 0$.

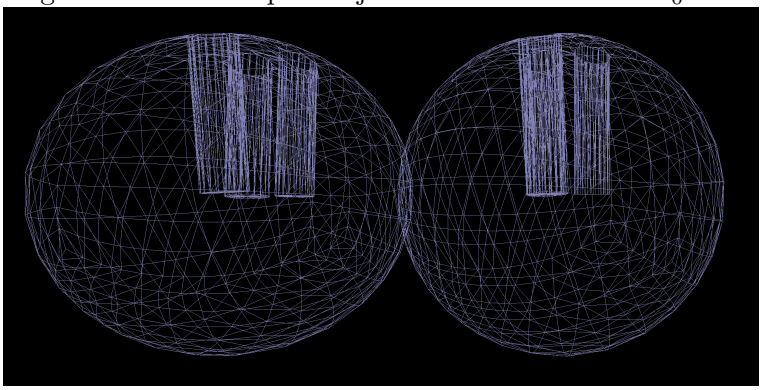
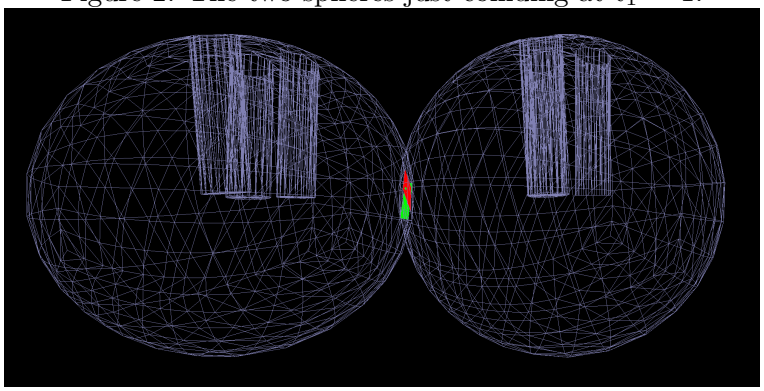


Figure 2: The two spheres just colliding at $t_1 = 1$.



```
checkInbetween(data, h)
 Input: data is information about the frame at t1.
 Input: h is a heuristic factor.
 Output: a bool which is true if collision has occurred.

 Generate the inbetween state by applying h to data
 result = check if collision occurs at the inbetween state
 return result

calculateTime(data, threshold, N)
 Input: data is information about the frame at t1.
 Input: threshold is the minimum allowed time difference between
    two extrapolated inbetween frames.
 Input: N is the max number of times to be iterated to compute
    inbetween frames.
 Output: Approximate time of collision.
 t0 = 0
 t1 = 1
 Iterate for N times
  if (t1 - t0 < threshold)
   break
  var = checkInbetween(data, (t1 − t0)/2 )
  if (var)
   t1 = (t1 − t0)/2
  else
   t0 = (t1 − t0)/2
 time = (t0 + 1)/2
 return time
```