%% UPDATE updates positions, velocities and accelerations.

%

% Discussion:

%

% The time integration is fully parallel.

%

% A velocity Verlet algorithm is used for the updating.

%

% x(t+dt) = x(t) + v(t) \* dt + 0.5 \* a(t) \* dt \* dt

% v(t+dt) = v(t) + 0.5 \* ( a(t) + a(t+dt) ) \* dt

% a(t+dt) = f(t) / m

%

% Licensing:

%

% This code is distributed under the GNU LGPL license.

%

% Modified:

%

% 15 July 2008

%

% Author:

%

% Original FORTRAN90 version by Bill Magro.

% MATLAB version by John Burkardt.

%

% Parameters:

%

% Input, integer NP, the number of particles.

%

% Input, integer ND, the number of spatial dimensions.

%

% Input, real POS(ND,NP), the position of each particle.

%

% Input, real VEL(ND,NP), the velocity of each particle.

%

% Input, real F(ND,NP), the force on each particle.

%

% Input, real ACC(ND,NP), the acceleration of each

% particle.

%

% Input, real MASS, the mass of each particle.

%

% Input, real DT, the time step.

%

% Output, real POS(ND,NP), the updated position of each particle.

%

% Output, real VEL(ND,NP), the updated velocity of each particle.

%

% Output, real ACC(ND,NP), the updated acceleration of each

% particle.

%

 rmass = 1.0 / mass;

 pos(1:nd,1:np) = pos(1:nd,1:np) + vel(1:nd,1:np) \* dt ...

 + 0.5 \* acc(1:nd,1:np) \* dt \* dt;

 vel(1:nd,1:np) = vel(1:nd,1:np) ...

 + 0.5 \* dt \* ( f(1:nd,1:np) \* rmass + acc(1:nd,1:np) );

 acc(1:nd,1:np) = f(1:nd,1:np) \* rmass;

 return

end

This in c++ can make like it in matlab

void Integrate()

{

 // Initialize positions and velocities

 Initialize();

 // Compute the forces from the initial positions and velocities

 ComputeForce();

 for (int timestep = \_begin; timestep <= \_end; timestep++)

 {

 // Initial integration using the forces from the previous step, noting that velocities are advanced in half time step

 for (int i = 0; i < Nparticles; i++)

 {

 v[i] = v[i] + (f[i] / mass[i]) \* delta\_t / 2;

 x[i] = x[i] + v[i] \* delta\_t;

 }

 // Compute new forces from new positions

 ComputeForce();

 // Final integration, update the velocities by half time step using the new forces

 for (int i = 0; i < Nparticles; i++)

 v[i] = v[i] + (f[i] / mass[i]) \* delta\_t / 2;