EE/CE 6302 Microprocessor System

Final Project Report

Multiple Balls Collision Simulation and Evaluation

Guided by
Professor Myoungsoo Jung

Designed by
Binal Kamani bxk131030
Vinoth Ravady vxr120730

By signing this form, I certify that any work submitted is original and belongs to our group unless adequately referenced. I certify that I have not made available this work to any other students in class.

04/30/2014

Project Objectives:

Our main aim of project was to implement computation algorithm on launchpad for simulating two – dimensional kinetic behavior of rigid balls within square area bounded by non rigid boundaries by enabling communication between launchpad and PC.

MOTIVATION:

1. Understanding of
    - Arm Cortex M4 – Stellaris Launchpad
    - PC Serial Communication using any programming language
2. Real time computation and communication between a microprocessor and host

This concept is used in many applications, some such examples are listed below.

- Detection of impending collision in Automated Cars
- Simulation of black-body radiation
- Video games such as billiard balls

In project, we implemented two different modes namely Computation mode and Visulization mode which takes care of which operation to perform either to do the computation of input balls' x and y position or showing elastic ball collision movements on GUI screen.

TEAM ROLES:

PHASE	BINAL	VINOTH
0	Java GUI Setup, swing	Launchpad Setup - UART + Timer
1	Protocol Design - UART comm	
2	Implementation of host and target protocol	
3	Design and implementation of computational algo integrated host and target	
4	CPU Utilization	Optimization of code

Specification

A square area inside which balls simulation is shown is having non – rigid boundaries hence any ball which hits the boundary then instead of bouncing back it exits system. Here we have assumed that there is no energy loss during collision. Collisions are elastic.

Ball – Obstacle collision follows,

- Law of conservation of energy
- Law of conservation of momentum
- Laws of motion

Ball Parameters (initially),

- Initial position for all balls: 0
- Initial velocity for all balls: 0

Obstacle here is in form of line whose co-ordinates are provided in input text file. Based on line with having different slope values (0, 90, 135 etc) can be generated in GUI. There will be one source hole from which number of balls (specified in input text file) will appear with initial velocity set to 0.

Calculation of ball speed and position is done in Launchpad having Arm Cortex M4 which is sending this information to GUI via serial communication (RS - 232).

Detailed Steps

TECHNICAL APPROACH:

We divided our project work in terms of different 5 phases as shown below.

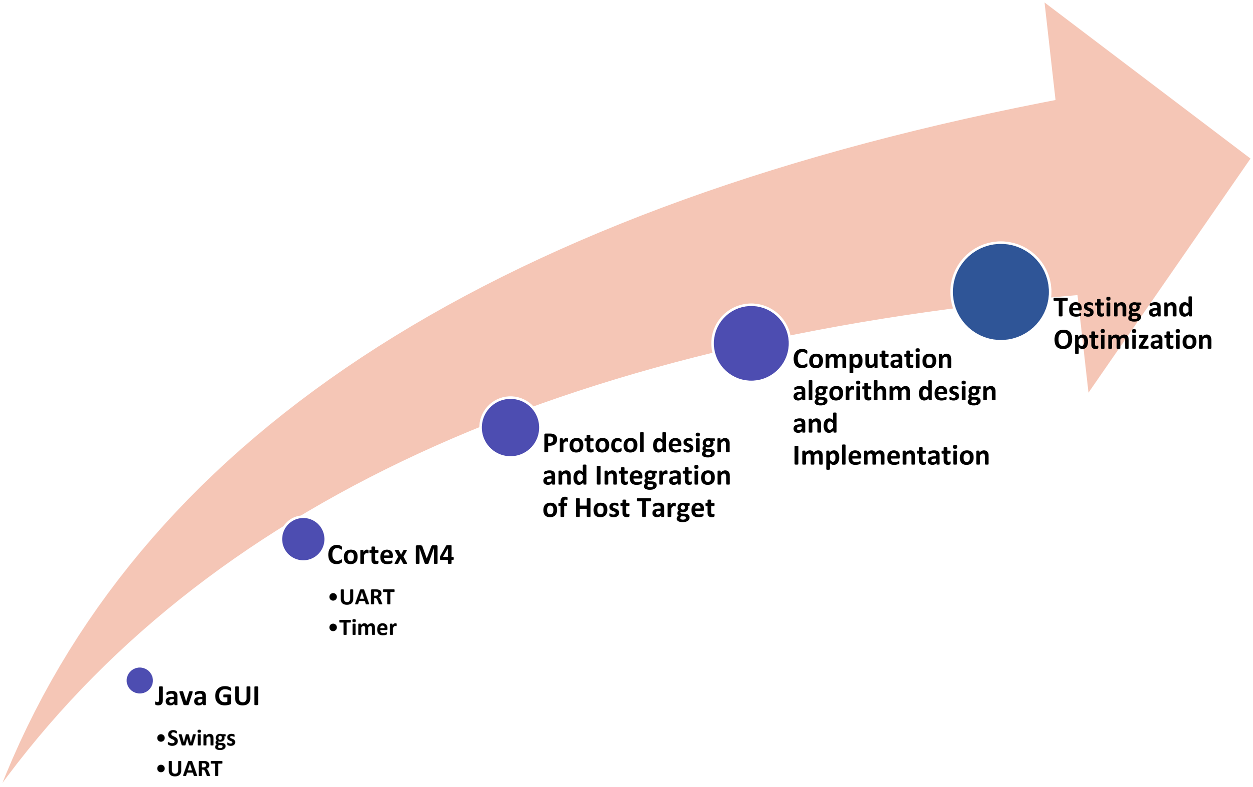
I. JAVA GUI – Work with JAVA Swings and basic test with serial communication

II. Target UART and Timer Setup – Configuring UART and Timer using CCS in Launchpad - ARM Cortex M4

III. Protocol Design for sending and receiving data – Defined protocol for serial communication on both side (Target and Host) in order to send and receive data efficiently

IV. Computational Algorithm – Figured out formulae and calculation part for ball position and speed to be shown on GUI

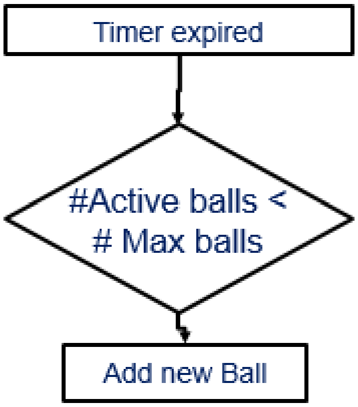
V. Testing and Optimization – Final testing and integration of algorithm as well as efficient optimization of algorithm


Java GUI
•Swings
•UART
Cortex M4
•UART
•Timer
Protocol design and Integration of Host Target
Computation algorithm design and Implementation
Testing and Optimization

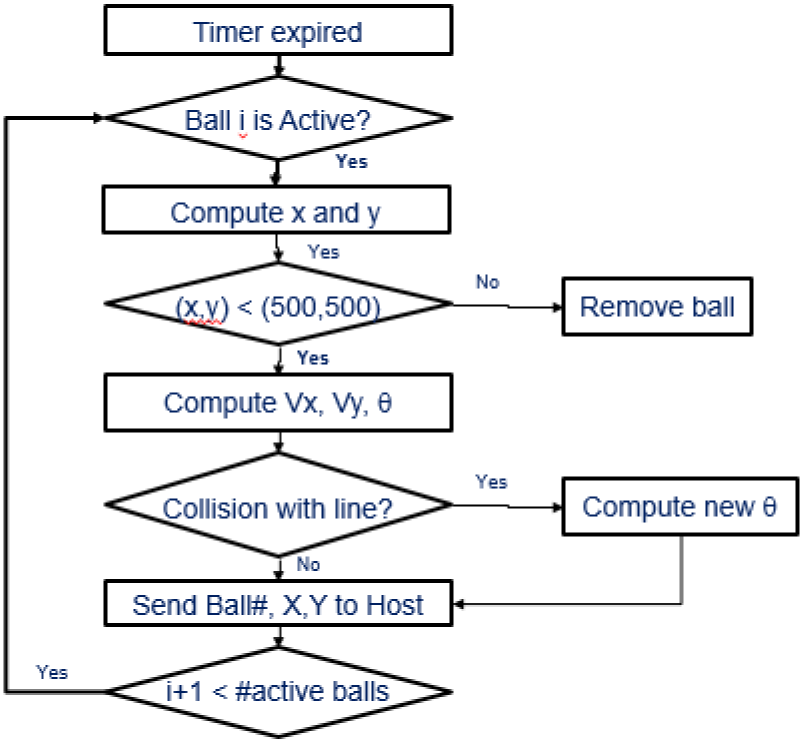
Main tasks to be done in Target are configuration of UART, Timer and computation algorithm implementation. Whereas in GUI, serial communication, use of java swing for different objects like balls and lines; real time update of balls position based on collision or free fall on screen, and providing user friendly interface with different facility like start, stop, resume and quit option for visualization in real time.

- ***Configuration of Timer and UART:***
    - We used two timers, one for generating delay time period of 40 ms which can be used to calculate all ball positions and speed each 40 ms. We chose this time duration basically due to 2 main reasons. One was to show visualization at least 24 times per second for better quality and secondly for new ball appearance after certain time period as specified in input text file.
    - Following are both timer's flow diagram. We have used Timer1A to generate time period for a new ball, when timer overflows and if number of balls generated till now is less than number of balls to be generated then a new ball is added with a different color to be distinguished properly.
    - Using Timer 0A, ball position and speed calculation is done each 40 ms. As shown in diagram we first check whether the ball is active or not i.e. if ball has exited the system then there is no need of computation. After this x and y positions are calculated and if position is in valid range then velocity for all active balls is calculated. Here we are using general formula for both free fall and trajectory motion, if the ball collides with line or obstacle then we are changing angle dynamically and sending this value to the GUI to be displayed in visualization mode.

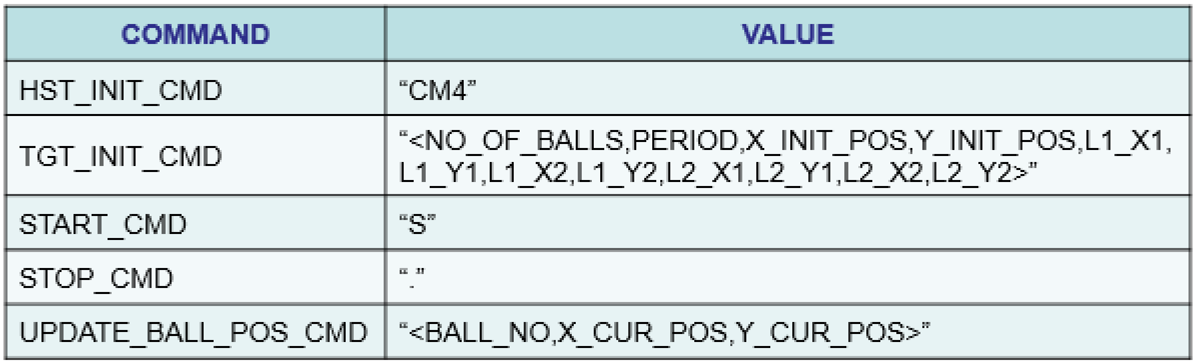
Timer 1A (period) Flow Diagram



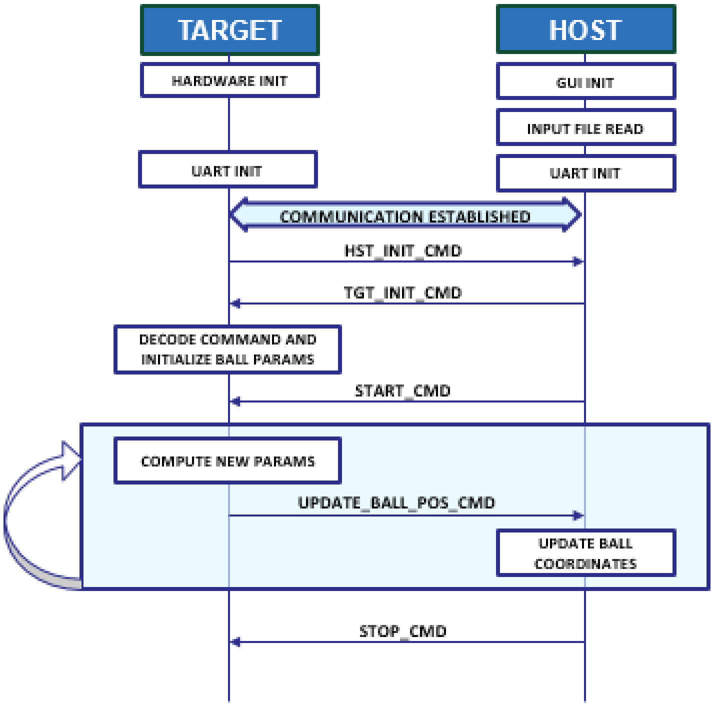
Timer 0A (40 ms) Flow Diagram



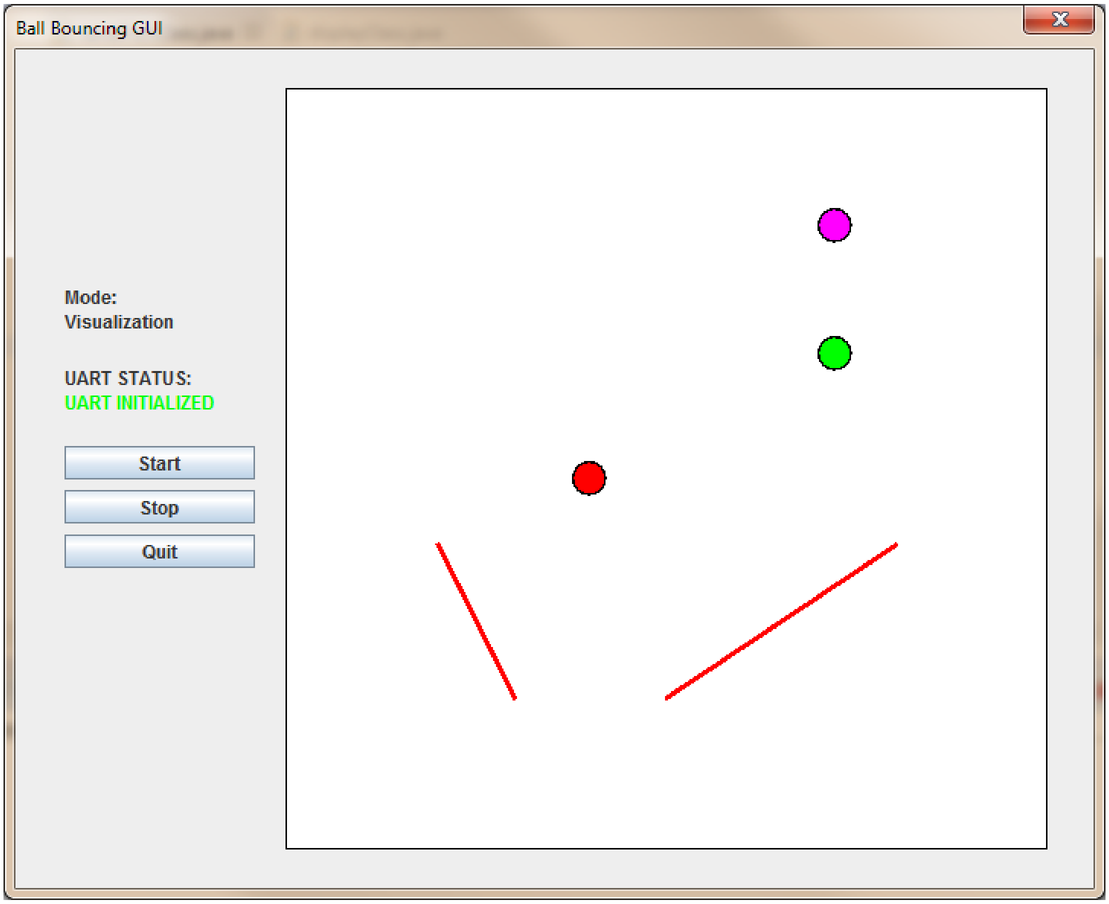
- For communicating in real time between Host and Target we selected **UART protocol** mainly because of its simplicity and features like Pin multiplexing, autoflow control.
- Below is the protocol commands and diagram of working.



COMMAND	VALUE
HST_INIT_CMD	"CM4"
TGT_INIT_CMD	"<NO_OF_BALLS,PERIOD,X_INIT_POS,Y_INIT_POS,L1_X1, L1_Y1,L1_X2,L1_Y2,L2_X1,L2_Y1,L2_X2,L2_Y2>"
START_CMD	"S"
STOP_CMD	"."
UPDATE_BALL_POS_CMD	"<BALL_NO,X_CUR_POS,Y_CUR_POS>"



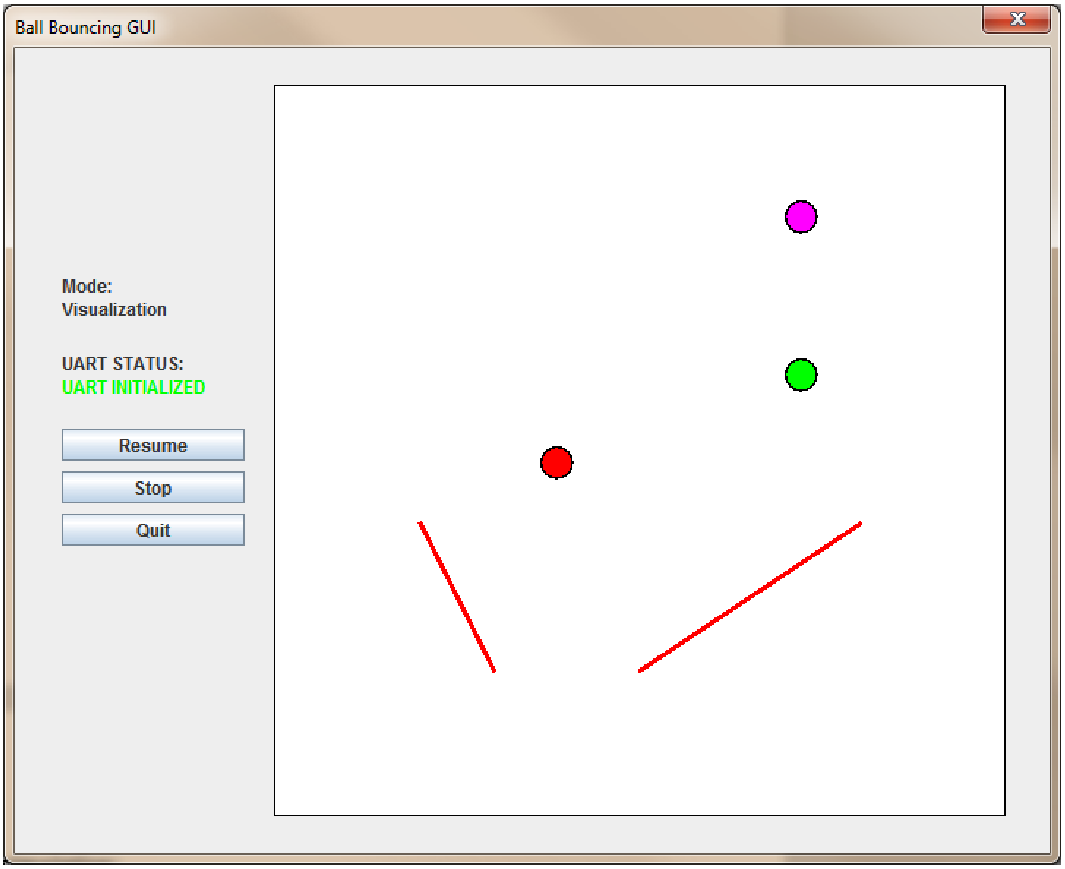
- As shown above after initialization on both Host and Target, host sends “HST_INIT_CMD” command to the target and target responds back by sending “TGT_INIT_CMD”.
- Balls initial parameter are set accordingly and when user clicks on Start button in GUI, it sends “START_CMD” to target after which computation of balls’ new parameter is done in real time. Positions of balls are updated correspondingly in GUI and once user clicks on Stop button, “STOP_CMD” is sent to target to stop calculation.
- Apart from this we have also included real time pause facility in our GUI, clicking on Start again after Stop button is clicked, timer will restart and ball positions are calculated from the point it was stopped.
- This is shown in pictures below,



Mode:
Visualization

UART STATUS:
UART INITIALIZED

- Fig A. GUI ball simulation in Visualization mode



Mode:
Visualization

UART STATUS:
UART INITIALIZED

- Fig B. Resuming and starting real time computation again

- ***Equations for ball simulation:***

  - We used equations for elastic collision as shown below,

Elastic Collision Equations

$$\boxed{m_1\vec{u}_1 + m_2\vec{u}_2 = m_1\vec{v}_1 + m_2\vec{v}_2}$$

$$\boxed{\frac{m_1\vec{u}_1}{2} + \frac{m_2\vec{u}_2}{2} = \frac{m_1\vec{v}_1}{2} + \frac{m_2\vec{v}_2}{2}}$$

  - For ball motion, we have used trajectory motion equations for both position and speed calculation of ball changing angle dynamically for free fall and projectile motion.

Ball Motion Equations

$$\boxed{\begin{aligned} v_x &= v_{0x} \\ x &= v_{0x}\,t \end{aligned}}$$

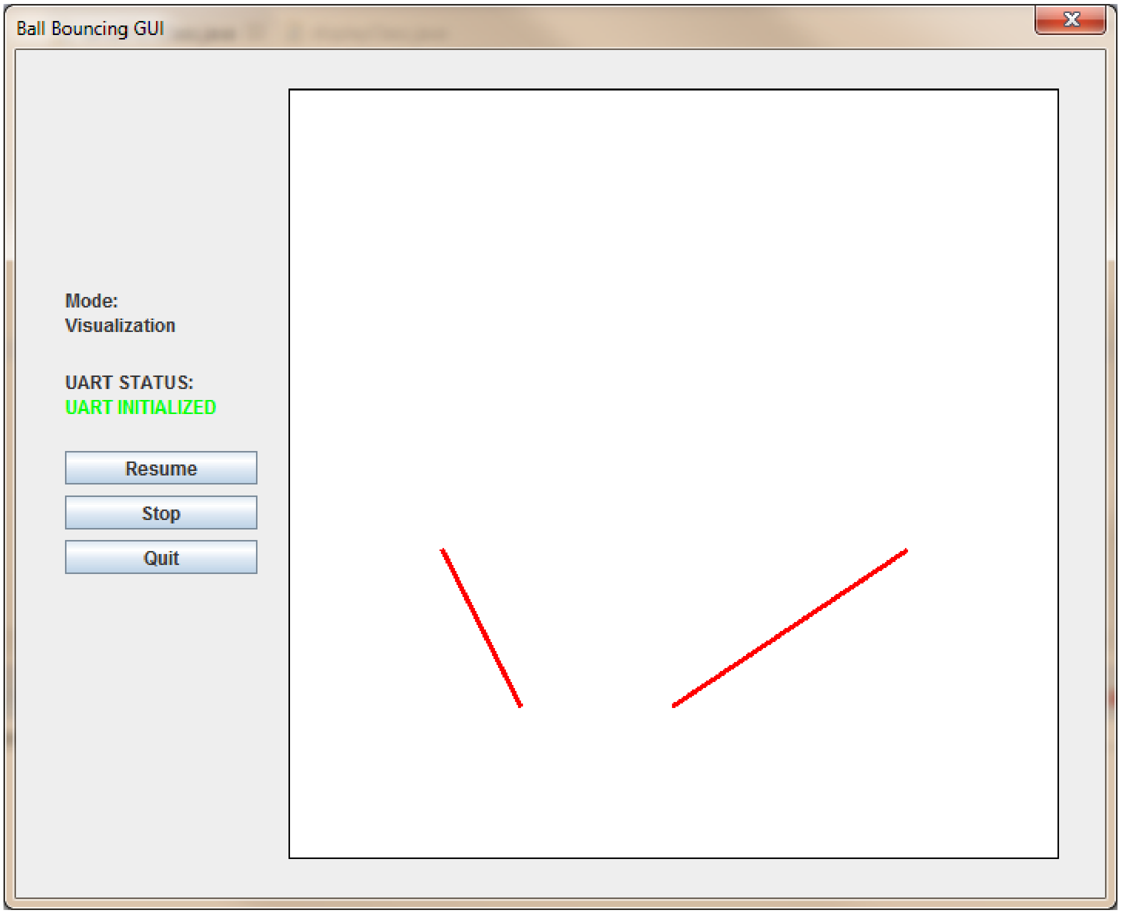
$$\boxed{\begin{aligned} v_y &= v_{0y} - gt & \text{Taking } g &= 9.8\ \text{m/s}^2 \\ y &= v_{0y}\,t - \tfrac{1}{2}g\,t^2 & &= 32.15\ \text{ft/s}^2 \end{aligned}}$$

Optimization and Results

We faced several problems during this project which are listed later. After resolving all these problem, we optimized our algorithm for maximum number of balls simulation as well as for efficient calculation of ball parameters.

Results and Extra Work:

- Dynamic real time calculation for both free fall and projectile motion using common position and velocity equation for both and updating angle.
- Improved on utilizing CPU for maximum number of balls(distinguished by different colors) that our algorithm can simulate : 165
- Improving GUI to provide user real time pause and resume facility apart from start and quit buttons.
- Displaying UART Status – “Initialized” or “NOT Initialized” in GUI to let user know about serial communication and initialization status of system.



Mode:
Visualization

UART STATUS:
UART INITIALIZED

Sample output:

1. ***Visualization Mode (Stopped after 10s)***

```
0 568 409
1 398 494
2 198 563
3 453 645
4 753 745
```

2. ***Computation Mode***
   a. ***Output (Stopped after 10s)***

```
0 900 447
1 900 644
```

   b. ***Output (Stopped after 10s)***

```
  0 0 0
 1 2 900
 2 2 900
 3 2 900
 4 2 900
 5 2 900
 6 2 900
 7 2 900
 8 2 900
 9 2 900
10 2 900
11 2 900
12 2 900
13 2 900
14 2 900
15 2 900
16 2 900
17 2 900
18 2 900
19 2 900
```

   c. ***Output (Stopped after 8s)***

```
   0 0 0
   1 0 0
   2 0 0
3 900 142
4 900 392
5 900 604
6 900 780
7 900 900
8 900 900
9 900 900
```

```
50 900 900
51 900 900
52 900 900
53 900 900
54 900 900
55 900 900
56 900 900
57 900 900
58 900 900
59 900 900
```

10 900 900	60 900 900
11 900 900	61 900 900
12 900 900	62 900 900
13 900 900	63 900 900
14 900 900	64 900 900
15 900 900	65 900 900
16 900 900	66 900 900
17 900 900	67 900 900
18 900 900	68 900 900
19 900 900	69 900 900
20 900 900	70 900 900
21 900 900	71 900 900
22 900 900	72 900 900
23 900 900	73 900 900
24 900 900	74 900 900
25 900 900	75 900 900
26 900 900	76 900 900
27 900 900	77 900 900
28 900 900	78 900 900
29 900 900	79 900 900
30 900 900	80 900 900
31 900 900	81 900 900
32 900 900	82 900 900
33 900 900	83 900 900
34 900 900	84 900 900
35 900 900	85 900 900
36 900 900	86 900 900
37 900 900	87 900 900
38 900 900	88 900 900
39 900 900	89 900 900
40 900 900	90 900 900
41 900 900	91 900 900
42 900 900	92 900 900
43 900 900	93 900 900
44 900 900	94 900 900
45 900 900	95 900 900
46 900 900	96 900 900
47 900 900	97 900 900
48 900 900	98 900 900
49 900 900	99 900 900

Lab Q/A

1. What problems have you encountered in this lab?

We faced several problems during this project as listed below,

- Finding a Serial Communication DLL
- Java JRE compatibility with Serial Communication DLL
- Decoding of host and target commands
- Making the design scalable for dynamic number of balls
- Detection of collision point between ball and line

2. What is the maximum number of balls you can simulate?

- 165

3. How you optimize your algorithm to shorten the calculation time?

Following steps we did to improve calculation time:

- Improving UART communication:
  By applying some changes in computation mode for sending data, we could improve upon calculation time. As only after calculating final position for balls only we are sending details to GUI for displaying it in text file.
- Less number of Branch:
  One of the reason for performance degradation is number of branches used in code as, we tried to use as less braches as we can to improve efficiency.
- Ball computation algorithm:
  For ball's free fall and projectile motion we used dynamic update of angle in speed and x,y position calculation equations which again helped making algorithm more efficient.

4. What is the Average percentage CPU utilization?

- Depending on number of balls and the time it takes to compute all ball position, speed; CPU utilization can be calculated. We checked for different number of balls and here are some results.

Number of Balls	CPU Utilization
1	6.20%
5	15.03%
10	32.57%
100	91.32%