

Carrom Tutor : Playing Strategies and Implementation

Dissertation

Submitted in partial fulfillment of the requirements
of the degree of

Master of Technology

by

Mayur Shashikant Katke

Roll Number: 123050069

Under the guidance of

Prof. Sridhar Iyer

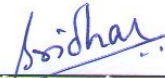


Department of Computer Science and Engineering
Indian Institute of Technology Bombay
Mumbai

2014

Dissertation Approval Certificate
Department of Computer Science and Engineering
Indian Institute of Technology, Bombay

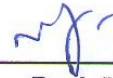
The dissertation entitled "Game Based Carrom Tutor", submitted by Mayur Shashikant Katke (Roll No: 123050069) is approved for the degree of Master of Technology in Computer Science and Engineering from Indian Institute of Technology, Bombay.



Prof. Sridhar Iyer
Dept. of CSE, IIT Bombay
Supervisor



Prof. Purushottam Kulkarni
Dept. of CSE, IIT Bombay
Internal Examiner



Prof. Vijay Raisinghani
Associate Dean (MPSTME) &
Professor, NMIMS
External Examiner



Prof. Abhay Karandiakar
Head, Dept. of Electrical Engineering, IIT Bombay
Chairperson

Place: IIT Bombay, Mumbai
Date: 19th June, 2014

Declaration

I declare that this written submission represents my ideas in my own words and where other's ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.



Signature

MAYUR SHASHIKANT KATKE

Name of Student

123050069

Roll number

19/06/2014

Date

Abstract

With the advent of new technologies and widespread use of Internet, usage of tutoring systems is increasing day by day. Online Tutors help in learning with flexibility and comfort to improve user's confidence. Game based Carrom Tutor is a tutoring environment aimed at teaching various Carrom skills and strategies to Carrom aspirants. We built two systems, Carrom Tutor 1.0 and Carrom Tutor 2.0. Carrom tutor 1.0 is a web based system which teaches carrom skills and test user's knowledge. Carrom Tutor 2.0 provides a game environment, using Blender 3D, for learning carrom skills. While designing these tutors we took into account perspectives of educational technology, Game-Based learning, and software and user interface design.

In this report, we discuss different Carrom skills and strategies. Then we describe the design, implementation and user experiments of Carrom Tutor 1.0. Then we present the detailed design of Carrom Tutor 2.0, followed by its evaluation. We demonstrate the effectiveness of our system by discussing the results of user experiment conducted with respect to another carrom application. Finally, we discuss the challenges faced while building the tutor and implications of our work for future development of the system.

Acknowledgements

I would like to thank Prof. Sridhar Iyer for his invaluable guidance. I also thank Mrinal Malick for his assistance and cooperation towards achieving the expected results in this project. I also thank Rwitajit Majumdar and Shitanshu Mishra greatly for mentoring me in right direction. I also thank Charan Ankam for helping me in identifying the Carrom skills and strategies. I would like to thank Prof. Sameer S. Sahasrabudhe and Nitin Ayer for their help.

Contents

Contents	2
List of Figures	5
1 Introduction	7
2 Background and Related Work	9
2.1 Carrom Skills	9
2.2 Carrom Strategies	17
2.3 Teaching Carrom	18
3 Carrom Tutor 1.0	21
3.1 Design of the tutor	21
3.1.1 Educational Technology Perspective	21
3.1.2 Design Perspective	22
3.1.3 Architectural view of the Tutor	23
3.2 Implementation	24
3.2.1 Macromedia Flash MX - Trial version	25
3.2.2 Html	25

3.2.3	CSS	26
3.2.4	JavaScript	26
3.3	Demonstration	28
3.4	User Experiments	30
3.4.1	Sample	32
3.4.2	Data Collection Methodology	32
3.4.3	Data Analysis and Results	33
3.5	Challenges	34
4	Motivation for Carrom Tutor 2.0	36
5	Design of Carrom Tutor 2.0	37
5.1	Educational Technology Perspective	37
5.2	Game-based learning Perspective	39
5.3	Design Perspective	39
5.4	Architectural view of Carrom Tutor 2.0	40
6	Implementation	41
6.1	Modelling	41
6.2	Logic Editor	42
6.3	Python Scripting	45
7	Demonstration	50
8	User Experiments	54

8.1	Sample	54
8.2	Data Collection Methodology	54
8.3	Data Analysis and Results	55
9	Challenges	57
10	Conclusion and Future Work	59
10.1	Conclusion	59
10.2	Future Work	59
	Bibliography	61

List of Figures

2.1	Straight Shot.	10
2.2	Normal Cut.	10
2.3	Straight Cut.	11
2.4	Negative Cut.	11
2.5	Doubling.	12
2.6	Punch.	12
2.7	Press.	12
2.8	Rebound.	13
2.9	Coin to coin Deflection.	13
2.10	Striker's Deflection.	14
2.11	Connection.	14
2.12	Cut Return.	15
2.13	Double Touch.	15
2.14	Follow.	16
2.15	Coin on Baseline.	16
2.16	Rolling of Striker.	17

3.1	Activity Diagram	23
3.2	Index page	29
3.3	Demo page	30
3.4	Exercise page	31
3.5	Recall exercise page	31
3.6	Exercise 2	32
3.7	SUS Feedback	34
6.1	User Interface of Blender Game Engine	42
6.2	Intra-scene object dependencies	44
6.3	Global variables used in intermediate scene	44
6.4	Logic bricks for playing video and loading practice exercise	45
7.1	Start screen	50
7.2	Second screen	51
7.3	Tutorial screen	52
7.4	Complex exercise	53
8.1	SUS Feedback for tutor	56

Chapter 1

Introduction

Game Tutors help in learning a new game and improving ones skill sets in all aspects. Tutor provides expertise, experience and inspiration for learning. For learning new things, necessity of a tutor becomes prominent. *Game based Carrom Tutor* is an initiative to teach various Carrom skills and strategies to Carrom aspirants. There is no such system available on the internet for carrom, but there are many Carrom games on the web. So this project is aimed at filling the absence of a good Carrom Tutor. The main objective of Game based Carrom Tutor is to improve ones Carrom related skills which generally cannot be learnt without any proper guidance. It provides a tutoring environment for teaching Carrom skills ranging from basic to advanced and exercises for testing user's skills.

The best way to learn carrom skills and strategies is to watch someone/expert playing carrom. For learning carrom, there is need of some expert assistance but getting professional support is not always possible. There is not a single carrom game or application available on internet which provides opportunities to learn carrom skills and strategies to users. *Game based Carrom Tutor* fits well in this context and it aims at teaching almost all Carrom skills to users. Model of Game based learning was followed while designing the overall content of this tutor. For an improved learning experience, the teaching contents of this tutor are inter weaved with the gaming environment. This fusion of game like environment and learning content naturally encourages users to repeatedly perform an action till the desired level of proficiency is achieved.

Carrom Tutor 1.0 is the web based system implemented using *HTML*, *CSS* and *Java Script*. It teaches carrom skills to users by demonstration of skills and test user's learning. This system does not provide flexibility and full control to users while playing the exercises.

Carrom Tutor 2.0 is a game which provides opportunities to learn various carrom skills and strategies to users. It is implemented in *Blender Game Engine*. Python scripts and Logic editor are two major components of implementation in Blender. Tutor has two main parts to help users in the process of learning. First part provides demonstration of various Carrom skills through videos. Two practice exercises are given to users for playing the same skill after video. Users are then provided complex exercises wherein they can apply previously learnt skills in game like environment. Users can play shot in given board situation to get maximum points and test skills. Evaluation is done based on user's shot selection and strategies used.

Various carrom skills and strategies are discussed in chapter 2. Carrom teaching methods are also described in this chapter. Design, implementation and user experiments of Carrom Tutor 1.0 are discussed in chapter 3. Chapter 4 mentions the motivation for building Carrom Tutor 2.0 and it's overview. Design of Carrom Tutor 2.0 is discussed in chapter 5 and Chapter 6 describes the implementation of Carrom Tutor 2.0 in blender game engine. Results of user experiments conducted with number of users are described in chapter 8. Challenges faced while building the Carrom Tutor 2.0 are discussed in chapter 9.

As scope of this project is very large, it has been started as a joint project. More related information about this project can be found in dissertation written by Mrinal Malick [10]. Some images used in demonstration of *Carrom Tutor 1.0* 3.3 and *Carrom Tutor 2.0* 7 are same in both dissertations.

Chapter 2

Background and Related Work

Teaching of particular concept, fact or procedure associated with any specialized area requires expert knowledge in that field. This knowledge in educational technology is called as *Domain Knowledge*. Domain knowledge is necessary for teaching skills to novices but with that application of those skills to solve real world problems and proper structuring of the domain knowledge is very important in order to facilitate learning. For teaching Carrom skills and strategies almost all skills from basic to advanced are found and arranged in increasing level of difficulty.

Before starting to look at the Carrom skills user should know about some basics about taking a grip on striker and releasing it. There are various shooting styles for Carrom. Holding the striker with your hand and releasing it to aim is very important. You can use any style of grip for playing. Mostly the ‘Straight Grip’ with index finger is used for playing Carrom board game. Some players use scissor grip for playing. Thumb and scissor grip are used for playing back-shots or coins below base line.

2.1 Carrom Skills

Carrom skills with increasing order of difficulty are discussed below [11]. Whenever necessary there is a diagram showing the coin positions on board and textual explanation for playing that shot is also provided. To avoid confusion, coin positions in diagram for all the skills are shown considering that player’s coin are black and opponent’s coin are white. Textual explanation is divided in two parts i.e. *Coin Positions* and *How to play*.

1. Hitting the coin : Initially try to aim the coin and hit it by striker with comfortable grip.

2. Hitting the coin with direction and force : Once you can hit the aimed coin then try to pocket it by giving it proper direction and hit it with required force.

3. Straight Shot :

Coin Positions : There are no obstacles for coin in traveling line to the pocket and the base line is free.

How to play : Place the striker on the straight line passing through coin and the pocket in which you want to pot it and hit coin at point with slow speed towards pocket.



Figure 2.1: Straight Shot.

4. Normal Cut :

Coin Positions : There are no obstacles for coin in traveling line to the pocket but some other carrom men are restricting the striker placing.

How to play : Place striker in small angle with coin and hit the coin at a particular point to pot it.



Figure 2.2: Normal Cut.

5. Straight Cut :

Coin Positions : There are no obstacles for coin in traveling line to the pocket but some other carrom men are restricting the striker placing.

How to play : Striker is placed in straight line(or perpendicular line) to coin and released to cut the coin towards pocket.



Figure 2.3: Straight Cut.

6. Negative Cut :

Coin Positions : There are no obstacles for coin in traveling line to the pocket but player wants to direct his striker towards some desired place on board for releasing his coins/fixing opponents easy coins.

How to play : To pocket a coin in right/left forward pocket the striker is placed at extreme right/left of base line respectively and away from straight line(perpendicular) of the coin to cut the coin towards pocket with medium speed.



Figure 2.4: Negative Cut.

7. Doubling :

Coin Positions : The coin is nearer to the base cushion and difficult to cut in pocket or there are some hindrances in playing the coin straight to pocket.

How to play : Double the coin in base pocket by striking it in such a way that it goes and strikes base cushion and directs to the desired pocket.



Figure 2.5: Doubling.

8. Punch :

Coin Positions : Coin is touching the cushion or there is very small gap between coin and the cushion.

How to play : Striker should hit the quarter portion of coin which is opposite to the pocket in which you want to pocket the coin with medium speed. The point of contact of coin and striker and force of the shot are very important to pot the coin.



Figure 2.6: Punch.

9. Press :

Coin Positions : Two coins on the board are parallel to forward base cushion and it seems that no coin can be pocketed. But left/right coin can be pocketed using this skill.

How to play : If you want to pot the left coin then release the striker such that it pushes the right coin forward without touching the left coin and then



Figure 2.7: Press.

striker on its way cuts the left coin towards left forward base pocket.

10. Rebound :

Coin Positions : The coin is slightly above the players base line and below right/left base line and it can not be played using thumb/scissor.

How to play : Hit the striker on forward base cushion with some force such that it returns and hits the coin to pot it in right/left base pocket.



Figure 2.8: Rebound.

11. Coin to Coin Deflection :

Coin Positions : The coin is interrupted by other carrom men on its way to the pocket or the player wants to disturb other coin which she is going to use for pocketing the coin.

How to play : The coin can be played to hit another coin first and then it gets deflected towards the pocket. For this shot you need to be very familiar with weight of the coins and deflection it will take after touching another coin.



Figure 2.9: Coin to coin Deflection.

12. Striker's Deflection :

Coin Positions : There are some obstacles on the traveling line of striker towards coin and striker can not hit the coin directly. Or the coin is near to pocket and very easy to pot but the player wants to take advantage of this and remove the obstacles for his/her other coins or fix opponents coins.

How to play : Striker is released to touch the obstacle coin and then get deflected towards the coin which we want to pocket. For playing these shots you need to be very familiar with weight of the striker and deflection it will get after touching the coin.



Figure 2.10: Striker's Deflection.

13. Connection :

Coin Positions : When two or more coins are in a row and you want to pot the last one, then this skill can be used. When coins are not in exact row then also this shot can be played. Sometimes you can use this skill to pot your coin by using opponents coin and making it difficult to play for her.

How to play : Hit the immediate coin in a way that it will strike other coin at position to pocket it. When there are more than two coins involved in connection then visualize the path of coins correctly and then play.



Figure 2.11: Connection.

14. Cut Return :

Coin Positions : One coin is near to the forward base lines and the other is slightly above the players base line and below right/left base line similar to that one in rebound shot. This shot can be played to pocket one/two coins or to pocket one coin and release the other. Sometimes it can be played to pocket our coin and make opponents coin difficult for her.

How to play : Cut the coin which is near to forward base line such that striker deflects towards forward base cushion and rebounds on other coin to pocket it. Here you use striker's deflection and rebound skill together.



Figure 2.12: Cut Return.

15. Double Touch :

Coin Positions : This technique is executed in small surface area near the pocket and with cushion. The coin is slightly above or below the players base circle and near to left/right base cushion. This coin can not be attempted with cut.

How to play : The coin is hit slightly in pockets direction towards left/right base cushion such that it will return back towards striker but slightly down respective to the previous position. When it touches striker second time it gets directed towards pocket. Coin gets two touches from striker to pot therefore it is called double touch.



Figure 2.13: Double Touch.

16. Follow :

Coin Positions : Opponents coin is near to the cushion and blocking player's coin which is behind it. Distance between the cushion and player's coin is more than that of opponent's coin.

How to play : Hit the coin which you want to pot with force such that striker will follow the coin. This coin hits opponent's coin which is a block and that coin moves outward by touching to cushion because of the force. Once opponent's coin moved outward striker which was following the coin hits the coin again towards the pocket. Even if striker doesn't hit the coin perfectly towards pocket coin gets the support from opponent's coin to move towards pocket.



Figure 2.14: Follow.

17. Playing coin on the Baseline :

Coin Positions : Coin is touching the upper base line and is very difficult to pot using thumb/scissor. Sometimes this shot is played with maximum force to pot the coin and disturb the other board so as to release coins for our next turn or to remove the obstacles for our other coins.

How to play : Make sure that your elbow is inside corner boundary and then cut the coin towards pocket with required force.



Figure 2.15: Coin on Baseline.

18. Rolling of Striker :

Coin Positions : For this shot situation is similar to that of punch but only the coin is not near to pocket, it is somewhere in the middle and there are many coins(our) in opposite side of the pocket in which we want to pot the coin.

How to play : Punch the coin towards open pocket with maximum force such that the coin will get potted and striker will move very fast in opposite direction where many coins are randomly placed. Striker moving very fast in opposite direction takes other coins towards pocket or sometimes pots one of those.



Figure 2.16: Rolling of Striker.

2.2 Carrom Strategies

There are no hard and fast set of strategies to follow while playing a carrom board game but there are some set of strategies that you should follow to make your game better [11]. Some of the strategies for the board game of singles and doubles are different. Few of them are explained below.

- **Break :** You can open a game in any way you want. But mostly in singles players break the frame in backward direction by hitting the forward base line and in return path striker hits the white coin at the end of diagonal of frame. This should played with maximum force and it gives good results. Coins near to your base line are very difficult to play for opponent.
- **Give Importance to Queen :** In a board of 29 points, Queen is assigned 5 points and in a board of 25 points it has 3 points. So pocketing queen is very important for getting more points and winning a game. Always check queen position before playing your shot. If it is easy for you to pocket it, then pocket the queen and cover. If it is easy for your opponent then either block the pocket or play queen and make it difficult for opponent after pocketing your easy coins.

- **Easy Cover** : After pocketing queen always take most easy coin as a cover because points of queen always play significant role in winning a game.
- **Check Blocks** : Check whether coin you are playing is blocking your opponent's coins (obstacle for opponent) before playing. If it is a block then don't play it. It prevents opponent from finishing her game and creates a mental pressure on her.
- **Coins and shots for breaking** : Suppose, there are many coins on the board and most of them are concentrated on small portion of board and one coin is on baseline. Here, cut shot can be played with force to pocket that coin and striker will be directed towards other coins to loose them. Similarly Doubling and Cut Return can be applied in different situations for breaking the group of coins or dragging coins towards baseline.
- **Negative Game** : Sometimes player don't have any coin which can be pocketed. In such situation, dragging maximum coins towards players baseline is the best option. If less number of coins are there on board then player can directly hit opponents coin and make that difficult for her. Playing opponents coin directly for fixing it is called negative game. When opponent is about to finish her game then this technique is very useful.
- **Safe Game** : When opponent is on the way of winning game, then player should not try difficult shots for pocketing coins instead she should pocket all easy coins and then try to make opponent's game difficult so that next turn can be obtained.
- **Game pass** : After trying all options when the player can't win the game then priority should be given to decreasing the points of opponent. If opponent is getting enough points to make game of twenty nine or twenty five points then pocket as many coins as possible to reduce the points and bring opponent's points less than twenty nine or twenty five so that next frame can be played and a chance to get control of the game can be obtained.

2.3 Teaching Carrom

Players can learn basic and some intermediate level skills which are straightforward, on their own. But some intermediate and advanced skills are not known to an average player. These are just an application of basic skills but the possibility of such application of basic skills is not known to average players. The best way to learn advanced skills and

application of various skills in different ways is to watch someone/expert playing those shots. Only the idea of playing a shot should be conveyed to learners to teach them. Once they know the idea, they can apply that in board and practice more to play such difficult shots.

To convey the idea behind every skill, video of that particular skill is shown to learners and with this text explanation for that skill is also provided to them in the tutor. After going through this, learners will be atleast able to decide which skill should be played in which situation.

Chess is also a board game. It is very much different than Carrom but the learning and teaching theory of one game can be applied in another. Chess teaching manual from Chess Federation of Canada[12] explains each move to user with text explanation and diagrammatic representation. Initially basics of the game were explained to learners and then each strategy of playing a move in a particular situation is explained in detail with diagrams. Also some situation of board is presented to learners and they are asked to answer proper move in that situation. Explanation for the correct answers is given to learners in step by step manner considering all possibilities.

Similar pattern is followed in the tutor for teaching carrom basics, skills and strategies. Tutor consists of animation of each skill ranging from basic to advanced with text explanation. Some board situations are also provided to learners and they are asked to play their shots in that situation with the purpose of winning the game and getting more points. This topic is discussed in detail in chapter 5.

Teaching Carrom skills to users can be done in different ways. One can learn playing Carrom in following common ways.

- The best way for teaching Carrom skills to new players is that experienced/skilled player should play/execute skill in front of them and tell them each and every details about that skill like where she should aim on coin, how much force is needed, proper visualization of the skill, which portion of the coin should be hit by striker etc. After this, novice can play the shot and skilled player can give her suggestions about the execution.

- Another way is to allow novice to play doubles board game with skilled players. In this, skilled partner can tell many different new shots and skills to novice for execution in order to win the game. Also the real time strategic thinking for game gets developed in novice.
- Carrom skills can be learned by watching skilled players board game. New player can watch different new shots and strategies in skilled player's game. New applications of basic skills and strategies and ideas of different shots can be learned in this way. Once player knows the idea and application of a particular skill, she can practice and play that skill.
- One more way of teaching Carrom skills to new players is to allow them to play more and more board point games. For winning the game, player tries to give her best. She tries many different skills and strategies to pocket more coins and block opponent's coins. In this way, player's game improves and she learns new skills and strategies. But this method is time consuming for learning of skills and not much effective.

Combination of two or more techniques can be used for teaching Carrom skills and strategies to aspirants. Most of the times, in practical scenario combination of techniques is used for teaching Carrom skills.

There are many Carrom games available on the internet. All of those allows player to play board game against the computer as opponent or another player can play as opponent in some games. From teaching-learning point of view, these Carrom games uses the last technique for teaching carrom skills. But it is not much effective as discussed before. There is no game which uses any of the first three teaching techniques or combination of those for teaching Carrom. So, for teaching Carrom Skills and strategies, there is no online Carrom tutor which uses any of the effective teaching strategies described before.

Chapter 3

Carrom Tutor 1.0

Carrom Tutor 1.0 is a web-based system and it uses effective teaching strategies discussed in chapter 2 for teaching carrom skills and strategies. Details of Carrom Tutor 1.0 are discussed in following sections of this chapter.

3.1 Design of the tutor

Tutor design is fixed after taking into account Educational perspective and software designing perspective. Educational perspective considers the learning theories and human learning styles for structuring the content and demonstrating it in such a way that it will help learning. Overall architecture of tutor is discussed in software designing approach. Following subsections discuss about these two in detail.

3.1.1 Educational Technology Perspective

Educational technology focuses upon learning process of a human being. Different educational technology principles were incorporated in Carrom Tutor 1.0 to facilitate learning. Some of these principles are described below.

1. ***Scaffolding*** : Scaffolding is a teaching strategy applied in Cognitive Apprenticeship in which teacher provides a support to help the student in performing a task. Teacher/Domain expert supports learner through suggestions or actual help in the task whenever needed [4].

In the tutor, after user has gone through all skill demos, a particular board position is given to her with the aim of finishing the game and gaining more points in exercises. When learner is involved in exercises and playing shots he is given some support from tutor. Whenever learner selects wrong striker position or wrong portion of the coin to hit, relevant message is given by tutor. Also when learner selects portion of the coin to hit which is near to the optimal one then message given by tutor gives a hint like “*You are very close.....*”. Design of the tutor uses Scaffolding as teaching technique in exercises.

2. ***Recall level questions and Understand and apply level exercises according to Bloom’s taxonomy :***

Human thinking is classified in six cognitive levels of complexity in Bloom’s taxonomy. First cognitive level is *Recall* in which previously learned or read facts are recognized and recalled by humans. Other cognitive levels of complexity are described in dissertation of Mrinal Malick [10]. Exercises in *Carrom Tutor 1.0* were designed by considering Bloom’s taxonomy. Some recall level exercises were given to user for step by step learning.

Cognitive model of human mind was considered while designing this tutor. Similarly modelling, sequencing were also applied. More details about these principles are given in section 5.1.

3.1.2 Design Perspective

Various options were considered for deciding each functionality in *Carrom Tutor 1.0*. All considered options, selected options for tutor and reasons behind choosing them are discussed in dissertation written by Mrinal Malick[10]. Initially, demos of all carrom skills are shown to users and then a complex board situation(exercise) is given to them for applying their learnt skills and strategies. In exercises, buttons are provided to users for selecting striker position, coin to play and sector of coin to hit. If user’s selected shot is correct, then animation for that shot is shown to them otherwise proper suggestions are given to them for playing the shot correctly.

3.1.3 Architectural view of the Tutor

Carrom Tutor 1.0 has two main components i.e. demos of carrom skills and exercises. There are many webpages created for showing demos and providing exercises to users. User inputs are taken with help of buttons provided on each webpage. Overall architecture of *Carrom Tutor 1.0* is described in this subsection.

Activity diagrams shows the sequence of activities which are happening when user interacts with any tool or undergoes through any process of the system. It displays the workflow from start point to end point with all the decision paths that exists in the system. Figure 3.1 shows the activity diagram of Carrom Tutor 1.0.

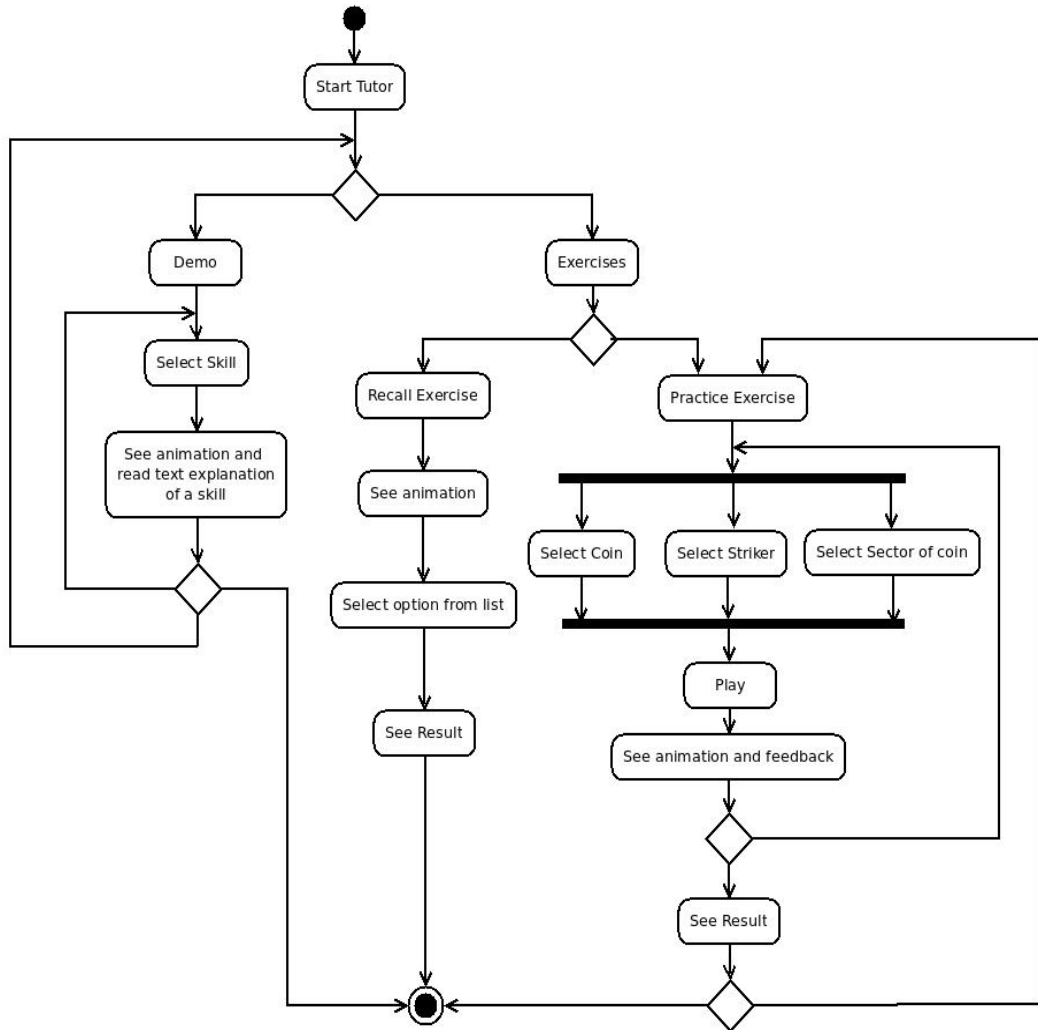


Figure 3.1: Activity Diagram

On start page of the tutor, four links are provided to users. These links are for demos, exercises, strategies and game rules. Links for strategies and game rules opens the corresponding webpages describing various strategies used in game and the game rules respectively. These two paths of user activity are not shown in activity diagram of *Carrom Tutor 1.0* 3.1. Other two paths of demos and exercises are shown in figure 3.1. In demo page, user can select the shot from menu and see animation of that shot. Text explanation about the shot and board situations in which the shot can be used is also given on the same page. In exercise section there are two types of exercises, Recall exercises and Practice exercises. Recall exercises comprises of an animation of some shot and user is asked to select proper option from the list which depicts the shot in the animation. In Practice exercises, a board situation is given to players and they have to pocket all their coins on the board to win game. Player first selects coin to be played, striker position and portion of the coin to hit by clicking on buttons. After this player can see the animation of shot which he has selected or proper message indicating the fault in selected shot will appear.

3.2 Implementation

After selection of various options for *Carro Tutor 1.0*, the overall design of the system was finalized as discussed in previous section. Implementation is very important part of any project because this phase actually builds the system to follow all architectural constraints decided in design phase. In implementation of tutor, much importance was given to finding a suitable platform and tool for building the tutor following the guidelines given by Paul[6].

Different technological platforms and tools were explored to find the most suitable option for implementation. Some of these options were selected for actual implementation and some of them were discarded because of mismatch with scope of the project. Unity game engine, OpenGL(Open Graphics Library) and JSP(Java Server pages) are some of those options which were not selected for tutor implementation.

Carrom Tutor 1.0 was implemented using *Html*, *CSS*, *Macromedia Flash MX - trial version* & *Java Script*. For creating the web pages and making them more interactive and attractive *Html* and *CSS* were used. All the internal logic related to user responses and

corresponding actions for responses was implemented in *JavaScript*.

3.2.1 Macromedia Flash MX - Trial version

Major application of *Macromedia Flash Mx* is to create *flash* animations. While making the animations, each object was placed in one layer. A timeline is provided to users for creating animation efficiently. Producing animations using this tool is easier, but the process of arranging the layers, timing the activities in those layers are some what tricky. All animations required for Carrom Tutor 1.0 were created using this tool.

3.2.2 Html

HTML is a standard markup language used to create web pages. Web pages for Carrom Tutor 1.0 were created using *HTML*. Three main pages i.e. Index page, Demo page and Exercise page were created for tutor. The *index.html* page is divided in five blocks. Each block with different name contains little information about it and has link to the corresponding page.

On *Exercise_Page.html* page, left portion has links provided for different exercises (*Recall & Practice* exercises). The middle portion has some general instructions about playing the exercise. In the top *div*, links to *Home & Demo* pages were given.

Clicking the particular *Exercise.i* redirects to the corresponding i^{th} exercise page namely *Exercise.i.html*. All the activities and options related to one exercise are handled in same page. In each exercise page, there are options provided to user for selecting a *Coin* and *Striker* on left side of the page. In middle *div*, at the top a circular image representing the twelve sectors with clickable buttons for each sector has kept for taking input from user. Below this image a text section is kept in form of *speech box* for displaying user's current selection for *Coin, Striker & Sector*. Depending on the chosen options of user, content of this speech box changes dynamically for showing user their currently selected options. Just after the speech box a *Play* button is given for showing animation of user's selected shot or to give proper feedback message. In the right *div*, animated gifs are shown to users according to their selected shot. Each exercise page contains links to *Home, Demo, Recall & Exercise_Page.html* at the top *div*.

Implementation of recall level exercise page (*Recall.html*) and demo page (*Demo.html*) is described in dissertation of Mrinal Malick [10].

3.2.3 CSS

Overall designing and modifications in the appearance of webpages is done using *CSS*. All the *CSS* properties are stored in a file named *mystyle.css* and this file is included in all *.html* pages to reflect those modified properties.

After that each list item is considered as a *html - button*. Then in *mystyle.css* different classes of different *buttons* have been created with different properties to distinguish between their actions. Later appropriate *Java Script* functions have been invoked using *onclick* property of *buttons*. *CSS* properties corresponding to *Play* button are given on next page.

Some more information about other type of buttons and properties of list buttons are described in [10].

3.2.4 JavaScript

Main part of the system implementation was completed by using *JavaScript*. User inputs were passed to *JavaScript* by using *onclick* property of buttons. According to those inputs web page contents are changed dynamically. In *Demo.html*, function *change(num)* was created to take an input value according to the user's chosen demo shot. This input value is given to *switch case* as choice for finding a match and according to that the content of the web page is changed. For changing the specific part of *html* page dynamically ***innerHTML*** property was used. This *innerHTML* is known as *HTML DOM* which provides the facility for accessing and manipulating *HTML* documents. The right *div* (where the animation of shots are shown) has been assigned *id = "content"* and the middle *div* has been assigned *id = "skill"*. Function "*change()*" changes the content of these *divs* to appropriate content (i.e., animation and text) depending on input.

In exercise pages, user selects the coin, striker and sector of the coin by clicking on buttons. Corresponding "*set()*" functions were invoked using *OnClick* property of but-

```

.button_Play{
  margin: 0;
  padding: 3px 5px 3px 5px;
  background-color: #0F6;
  border: 1px solid #0F6;
  width: 14em;
  height: 4em;
  color:#FFF;
  text-decoration:none;
  font-weight: bolder;
  font-family: Lucida Sans Unicode;
}

.button_Play:hover {
  background-color:#093;
  border: 1px solid #093;
  color:#FFF;
  transform: scale(1.2) translateZ(0);
  text-shadow: 0 1px 1px rgba(0,0,0,.3);
  -webkit-border-radius: .5em;
  -moz-border-radius: .5em;
  border-radius: .5em;
  -webkit-box-shadow: 0 1px 2px rgba(0,0,0,.2);
  -moz-box-shadow: 0 1px 2px rgba(0,0,0,.2);
  box-shadow: 0 1px 2px rgba(0,0,0,.2);
}

.button_Play:active{
  transform: scale(1.05) translateZ(0);
  background-color:#0F6;
  color:#093;
  border: 1px solid #0F6;
}

.button_Play:focus{
  background-color:#0F6;
  color:#093;
  border: 1px solid #0F6;
  -webkit-border-radius: .5em;
  -moz-border-radius: .5em;
  border-radius: .5em;
}

```

tons and the values of selected parameters are passed to that function for storing them in appropriate variables for future use by “*Play()*” function. “*set()*” function for Coin, Striker and Sector of coin are given below. In this code *num1*, *num2* & *num3* are those values which are passed to functions while calling.

```
var coin=0,striker=0,sector=0,points=0;
function setcoin(num1)
{
    coin = num1;
}

function setstriker(num2)
{
    striker=num2;
}

function setsector(num3)
{
    sector=num3;
}
```

In these pages the *id* of right *div* is “gif”. Clicking of *Play* button takes the current values stored in different variables and starts executing the *Play()* function. Some part of the *Play()* function is shown on next page. This function checks values of different variables and flags, then according to those values it examines different conditions given in the program to display appropriate animation or message. In this function *innerHTML* is used to change the animation of the rightmost *div* and *setTimeout()* function has been used to provide time delay wherever necessary. *Alert* boxes are used for displaying the messages.

3.3 Demonstration

Overview of *Carrom Tutor 1.0* is given in this section. User always starts the system with index page where different links are provided in the form of visual blocks. Users can see little description about the link given in each block once they hover upon them. The corresponding page will be loaded after clicking on the desired block. Five blocks on the index page corresponds to the links of exercises, demos, related documentations, Carrom rules & about the developers. Figure 3.2 shows the structure of index page.

As discussed in previous sections each page is divided in three parts. Demo page contains buttons for different types of demo shots on the left side. Once a button is pressed, corresponding animation is displayed at the right hand side of the page. Text explanation describing the selected skill is shown in middle part of the page for helping


```

function Play()
{
  Gif=document.getElementById("gif"),
  GifContent = Gif.innerHTML;
  .
  .
  .
  .
  if(coin==1 && striker==2 && sector==4 && flag2==2 && flag3==0)
  {
    choice=9;
  }
  if(coin==1 && striker==2 && sector!=4 && flag2==2 && flag3==0)
  {
    if(sector==3 || sector==5)
    {
      choice=100;
    }
    else
    {
      choice=26;
    }
  }
  .
  .
}

```

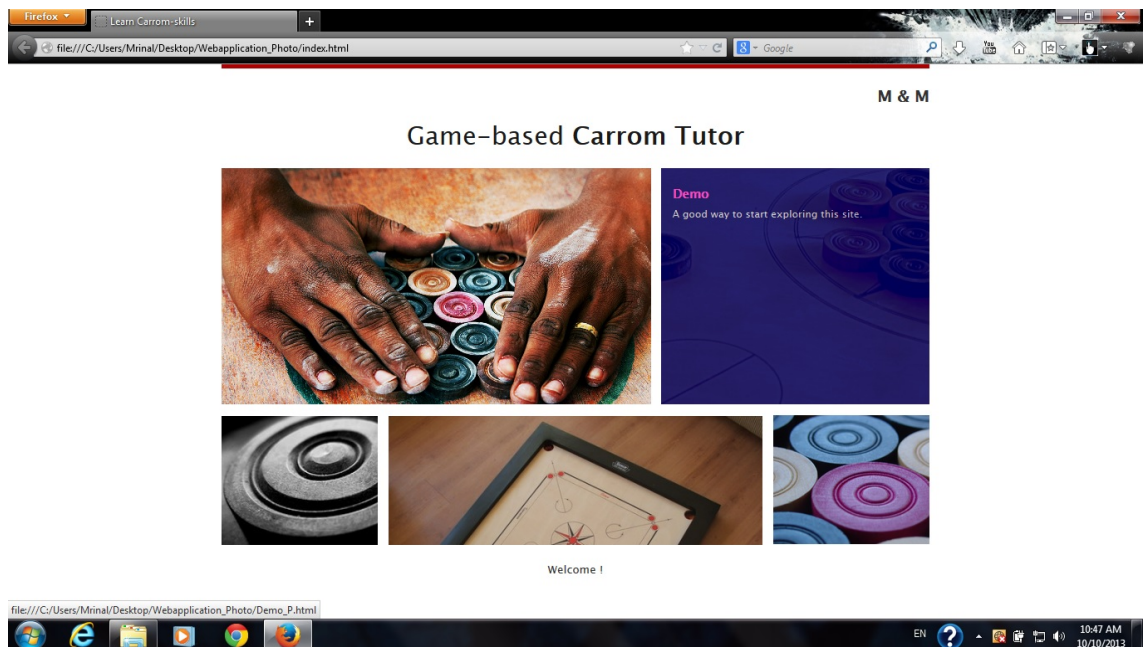


Figure 3.2: Index page

users to understand the skill properly. Demo page is shown in figure 3.3

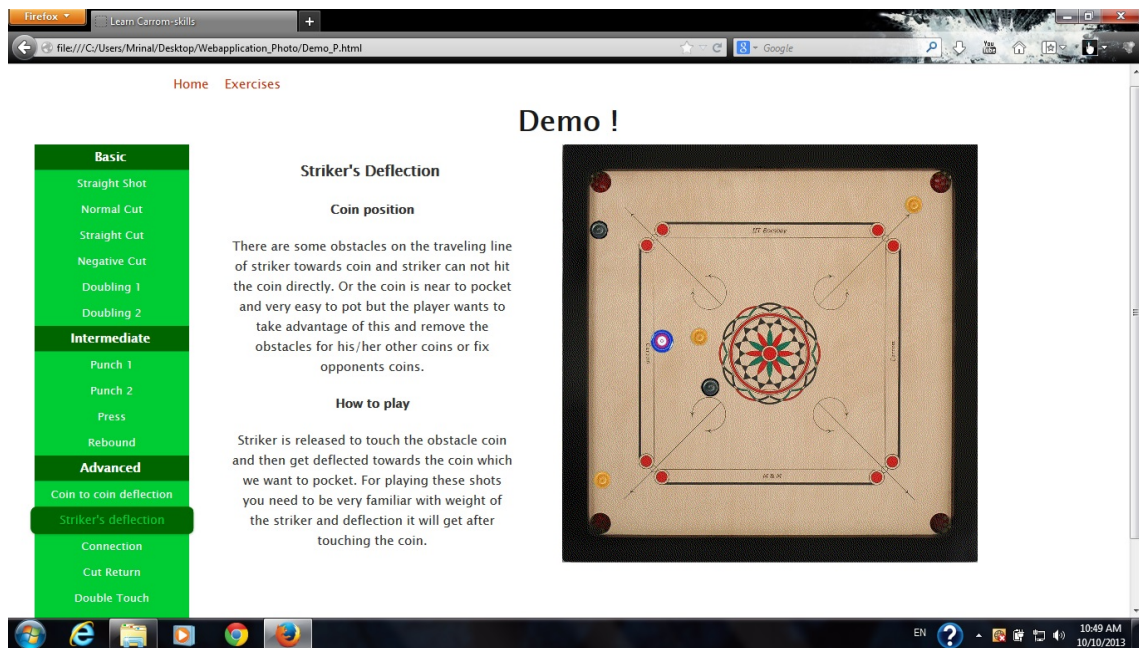


Figure 3.3: Demo page

Exercise_Page.html (Figure 3.4) shows the list of available exercises to the user. This list is shown at the left most portion of the page. Middle section contains some general instructions about the exercises, way of playing those exercises and explanation about the point system.

The general design of the page of recall level exercises is shown in figure 3.5. Exercises are listed in the left side of this page. The middle part of this page contains choices for each exercise. User has to choose from this set of given choices. If the selection is correct then a congratulation message will appear. Otherwise the user will be asked to try again.

Layout of Exercise page is shown is Figure 3.6. Left side of the page offers choices for coin and striker to the user. Middle portion contains choice for coin sector, speech box (showing the option chosen by user) and *Play* button.

3.4 User Experiments

This section discusses about the methods followed for conducting the user experiments of *Carrom Tutor 1.0*. Target audience, the way in which data is collected and analysis

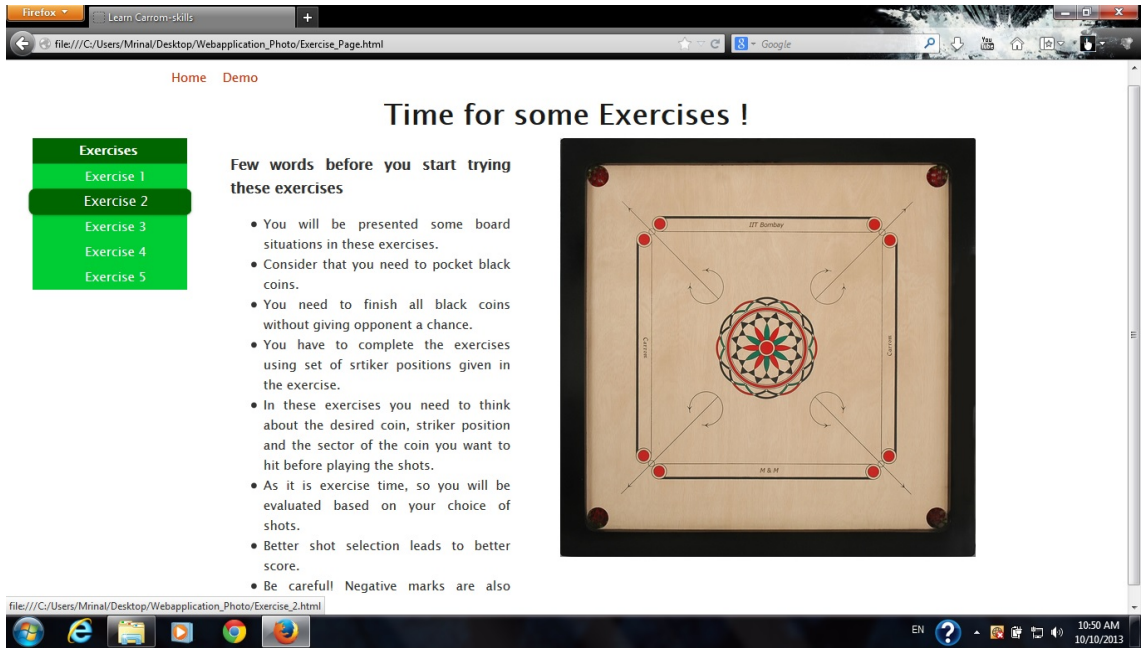


Figure 3.4: Exercise page

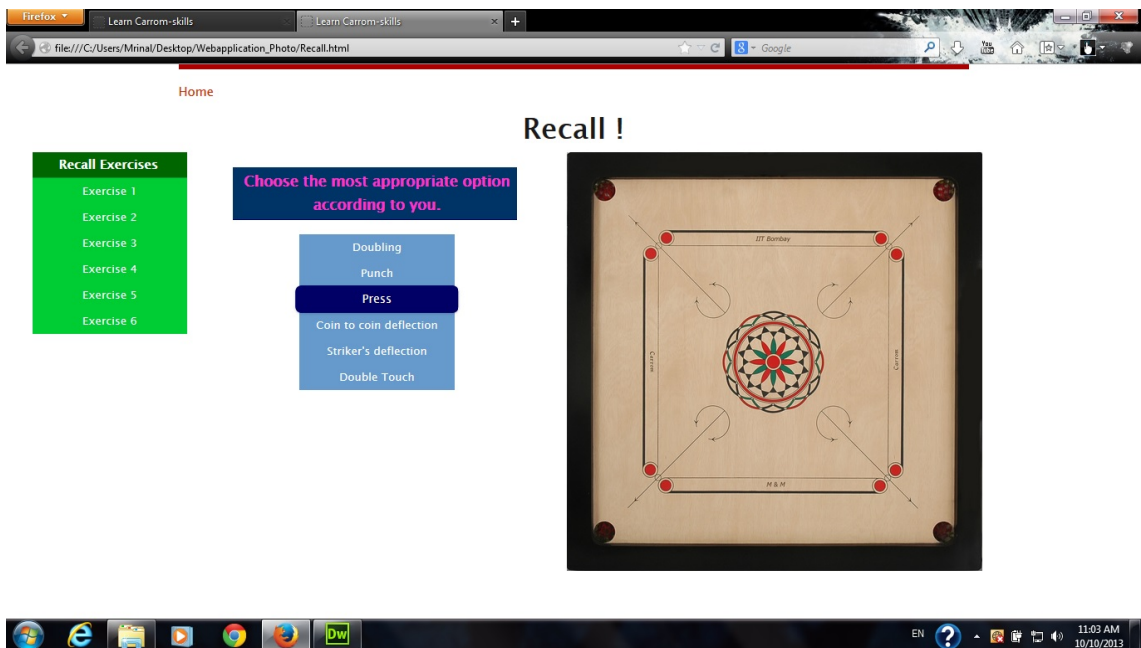


Figure 3.5: Recall exercise page

of collected data is described in this section.

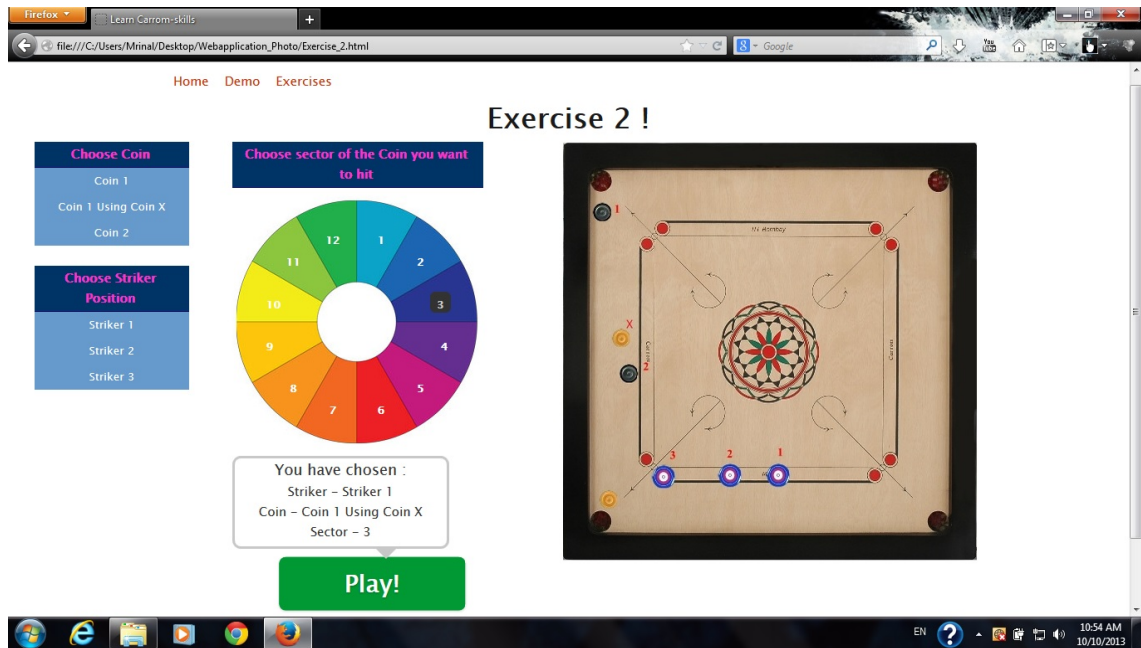


Figure 3.6: Exercise 2

3.4.1 Sample

Effectiveness of tutor will be decided by learning gain in users. User experiments are conducted with users who know about carrom board game or they should have played it some times. Data of the user experiments is collected from these samples. Seven users took part in this user experiment. These users were mixture of beginners and intermediates in Carrom.

3.4.2 Data Collection Methodology

Pre-post tests were included in user experiment. Before using the tutor some questions were asked to learners for identifying their interest. Also some of the exercise board situations were given to them and they were asked to tell which shots they will play in those exercises. This question was open-ended and asked for deciding the level of learner. Later user was allowed to use the system step by step i.e. first she sees the demos of all carrom skills and then exercises for practicing and testing their learning were presented to them.

Users were asked to give answers of the following questions after they have done with experiment.

- User's exposure to Carrom
- How many new Carrom skills you learned?
- How did the system helped you in learning Carrom?
- What is more difficult for you in learning Carrom?
 - Deciding Power
 - Aiming
 - Deciding Shot
 - Deciding Strategy
- Did the system helped you in learning above skill?

Data collected so far was for checking the learning gain in users. For usability testing of the system, SUS (System Usability Scale) analysis was done using *A quick and dirty usability scale* [3]. In this analysis, five point likert scale questions were asked to users. These questions were asked for checking accessibility, efficiency, effectiveness and attractiveness of the user interface of *Carrom Tutor 1.0*.

3.4.3 Data Analysis and Results

Usability testing of Carrom Tutor 1.0, SUS (System Usability Scale) analysis was done using *A quick and dirty usability scale* [3]. A set of ten questions were asked to the users for usability testing in SUS form. These questions were asked to test different parameters of usability of the system. Average percentage of SUS score for Carrom Tutor 1.0 is 77.14%. It shows that, it is a grade 'B' system according to usability scale [15]. Average responses given to each question asked in SUS analysis have been plotted in Figure 3.7, where decimal numbers 1,2,3,4,5 in user rating denote Strongly Disagree, Disagree, Neutral, Agree and Strongly Agree respectively. Detail explanation about SUS analysis, questions asked to users in this analysis and responses of users to those questions is given in dissertation written by Mrinal Malick [10].

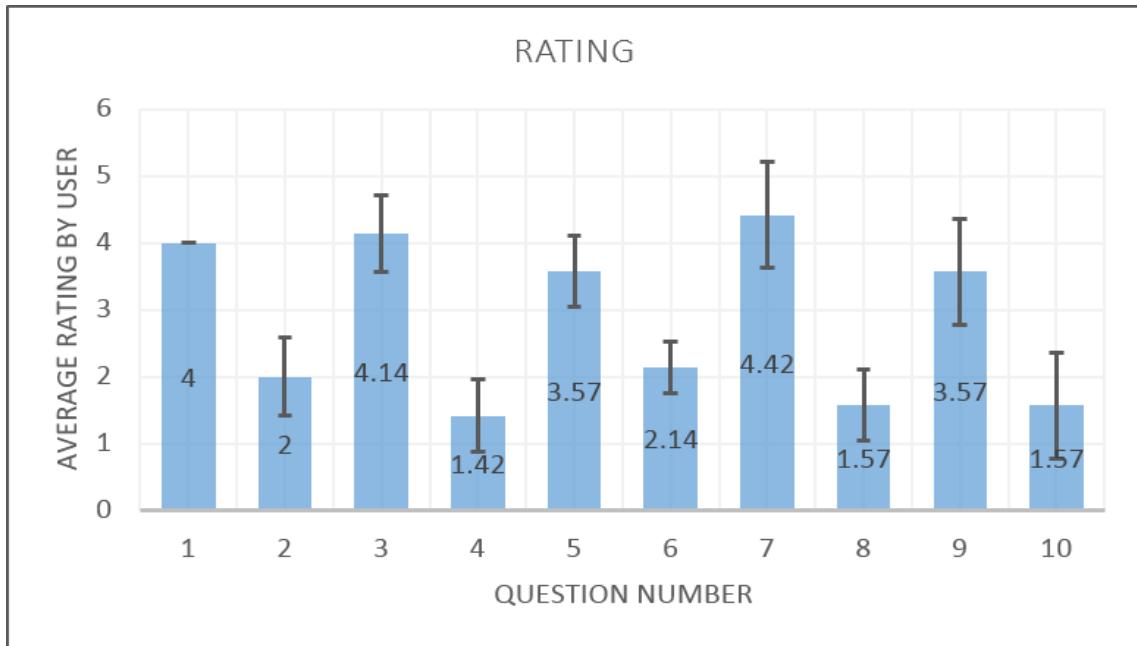


Figure 3.7: SUS Feedback

All users were asked to play complex exercises before and after going through demos of carrom skills. It was observed that users who were not able to play exercises before watching demos, were trying to apply optimal shots in those exercises after going through demos. Primary goal of Carrom Tutor 1.0 was to convey the ideas of playing different shots to users and that goal is achieved. Users have learned 2-5 new shots after watching demos. With answers to above mentioned questions, users have given many feedbacks for improving the system. Many of the users said that it would be better to show played shot to them even if it is wrong. Some of them were having some different shots in their mind and they suggested to allow users to place striker anywhere on baseline for playing. It was also suggested that instead of giving buttons for choosing striker and coin for playing, a game like interface should be provided for playing shots.

3.5 Challenges

Carrom Tutor 1.0 was implemented using HTML, CSS and JavaScript to meet all requirements decided in design of tutor. Different challenges and difficulties faced while implementing all features of tutor are discussed in this section.

The first exercise built using JSP was unnecessarily loading too many webpages for a single exercise. The complete system built using such approach will comprise of many webpages and much amount of redundant code. It was realized that designing of the tutor can be done using JavaScript with HTML and there is no need of using JSP for it. So, implementation of system was started using simple **HTML** web page designing, **CSS** and **JavaScript**. It took some time to learn JavaScript and CSS for this approach. Once some basic JavaScript programming was done actual implementation for Carrom Tutor 1.0 started. JavaScript was mainly used for changing the content of webpages dynamically depending on user's actions. To do this "*innerHTML*" property was used with "*getElementById()*" method to manipulate particular HTML element and to change the content of page without refreshing it. Separate *divs* with different ids were created for showing text part and animation of shot in demo page. After selection of a particular skill text explanation and the animation both get changed on same page. An exercise is created on a single page only. Multiple pages for each option were not created. This was a significant achievement over previous brute-force approach with JSP in which for each option separate webpage was created and loaded in the browser. In exercise pages, getting the values of different parameters for the user's shot and changing the page content according the combination of those values was somewhat difficult to implement and it took some time to do so.

Chapter 4

Motivation for Carrom Tutor 2.0

In *Carrom Tutor 1.0*, for watching the demonstration of skills and playing the exercises user need to click on buttons only. Full game like environment was not provided for playing exercises and learning skills. A demo of skill is shown to user by animated *gif* and text explanation about that skill was provided. Instructions and demo of skills were not provided in single flow to user. In exercises, user was allowed to place striker on pre-decided places on baseline from where the optimal shots for given coin positions can be played. Here, players thinking was restricted in particular area. Freedom to place striker anywhere on baseline was needed to be given to user in order to foster learning. Animated *gifs* were displayed for only the optimal shots selected by user. All possible outcomes of user's shot must be shown to users for making their learning effective. More flexibility and control were needed to be given to user and tutoring procedure must be improved. A good game environment was needed to be incorporated in exercises.

Blender Game Engine has been used for building Carrom Tutor 2.0. In this, instructions and demo of a skill is given to user in a single video and two practice exercises for same skill are provided. For exercises, full game like environment was created. To make it seem like a real playing experience, **3D** view is provided and it is presented from the players position for playing. This view is adjusted according to mouse movements of user. User is allowed to place striker anywhere on baseline and play desired shot. Here, user experiences actually what happens after playing a shot whether it is correct or not inside a real game. User can also select desired force for playing a shot in game. This is a game which teaches carrom skills and strategies to users.

Chapter 5

Design of Carrom Tutor 2.0

Features and requirements needed to be incorporated in *Carrom Tutor 2.0* were described in chapter 4. Design of tutor must be fixed so that it is consistent, fulfills the goals and meets all requirements. Tutor design was finalized after taking into account various perspectives like Educational perspective, Game-based learning and software designing perspective. Educational perspective considers the learning theories and human learning styles for structuring the content of tutor and demonstrating it in such a way that it will help learning. Desired features of Game based learning are added in Carrom Tutor 2.0 for making it more interesting and influencing. These features are discussed in Game based learning perspective.

5.1 Educational Technology Perspective

Our goal is to teach carrom skills and strategies ranging from basic to advanced to users through this tutoring environment. After going through this, user should be able to make a decision from the given board position that which shot and strategy she should apply at this point to get more advantage in game. Design of Carrom Tutor 2.0 is based on cognitive theory of learning so as to facilitate the learning of user. Tutor is constructed keeping in mind that how humans learn and what features we should use to promote learning. At the beginning, user is shown video of a carrom skill which gives instructions and a demo of skill. After this two practice exercises are provided to user for playing the same skill. Then the users are provided complex exercises and asked to apply their skills and shots to win the game and get points in each exercise. Tutor is built by exploiting

features of learning theories and principles. Following are some of them.

1. ***Modelling*** :

In Cognitive Apprenticeship, there are methods which give importance to providing a chance to students to observe, practice and learn expert strategies in a domain. Modeling is one of those teaching techniques. In this, students observe an expert performing a task with the purpose of understanding it and building a conceptual model of the process in mind. Once the process of achieving a task is clear in learner's mind then he practices it [4]. This technique of teaching is followed in tutor to teach carrom skills.

Initially, learner sees the video of a particular skill which shows them how an expert will play in that board situation. The text instructions provided elucidates the situations in which given skill can be implemented. After this, two practice exercises are provided to users for implementing the same skill. Finally, Complex exercises for implementing the skills and strategies are given to users.

2. ***Sequencing*** :

Cognitive Apprenticeship also emphasizes on the sequencing of learning activities so that the student learns expert skills step by step and their application in varied contexts. Depending on the domain, learning activities are sequenced to make understanding of skills simpler. One way to sequence the learning activities is that, they are arranged in increasing order of complexity. Activities are given to learner such that more and more expert skills and concepts are learned by student gradually and with increasing level of difficulty in each task [4].

When user is learning skills in tutor all skills are arranged in increasing level of difficulty. User can see any skill demo by clicking on button, but the menu on the screen displays the skills in three levels ie. Basic, Intermediate and Advanced. All skills are in increasing order of difficulty in menu so that user will learn them step by step and gradually move towards expert level of skill execution. Skills learned initially are used and applied in higher level skills.

3. ***Cognitive Model of the Mind*** : Human mind have sensory memory, working memory (limited capacity), and long-term memory (infinite capacity). Demos of carrom skills were displayed to users such that working memory of human mind is fully utilized by using separate channels for verbal and visual material. Some more description about cognitive model of mind and utilizing its full capacity in better manner for *Carrom Tutor 2.0* can be found in thesis written by Mrinal Malick [10].

5.2 Game-based learning Perspective

Features of *Game Based Learning* are discussed in dissertation written by Mrinal Malick [10]. For making game interesting, the user interface should be attractive. There should be some motivation in game for users. Influencing nature of the game forces user to repeat the exercises or quests given in the game over and over until desired level of satisfaction is attained. Reward system or points system can be added in the game for making it more exciting. Highest score for each exercise can be decided and user will be awarded points for his successful action. Such point system is added in the Carrom Tutor 2.0 and at the end user can compare her score with the highest one. If the expectation is not met user can try the exercise again. Game environment which is like real world scenario, influences users and results in effective learning. For adding this feature in system, **3D** environment is provided to users for playing exercises on carrom board.

5.3 Design Perspective

Each functionality and feature of *Carrom Tutor 2.0* was finalized after considering many options. User's comfort was considered for choosing the best option from them. Overall design of the tutor and its functionalities are described in this section.

Carrom skills are taught to user by a video and two practice exercises for a skill. Instructions about playing the shot and its demo are shown to user in single video. After this video, two practice exercises for the same skill are given to user. These practice exercises are simple and small. It will improve users learning and understanding of application of a particular skill. A set of complex exercises is also provided to user for testing their skills. These exercises are large and needs application of tricky shots to finish. It takes two or more shots to complete these exercises. According to the shot played to pocket a coin, user will get points. Score is displayed to user on upper right corner of screen.

Practice exercises and complex exercises are created in such a way that user can place striker on baseline by using left and right arrow key. Users are not allowed to place striker on half balls for playing. A red line is drawn to show the path of striker so that user can imagine the striker's collisions and its movements. Striker's direction can be changed by mouse movements. According to view adjusted and striker's direction, the red line

also rotates and shows striker's path every time. First left click will enable force gauge and fix the striker's position. Force gauge will be continuously moving up and down and it represents the amount of force applied on striker for playing a shot. Second left click applies the force proportional to slider's height on striker in desired direction. If the exercise is finished, necessary feedback and message is given to user. But if it is not finished correctly, same exercise is reloaded.

While playing exercises, users are provided a view of carrom board from the place where player sits for playing a game. This was done for making it like a real carrom playing experience for users. Camera is placed at a particular point to present this view and it is rotated according to user inputs. For deciding the amount of force applied on striker while playing a shot, many alternatives were considered. One of them was providing a force bar on which user can click and select amount of force before playing shot and then play. In many games similar to carrom, like pool, a force gauge is provided to users for deciding power of a shot. This force gauge continuously moves up and down and depending on users selection force is applied to play a shot. Same feature is used in Carrom Tutor 2.0 and for picking maximum and minimum force on slider easily, its movement is slowed at these two peaks.

Different properties of carrom board, striker and coins were fixed to make their movements and behavior same as the real ones. Friction and elasticity for these objects was fixed by trying different values. Values which gave more realistic movements of objects were finalized.

5.4 Architectural view of Carrom Tutor 2.0

Carrom Tutor 2.0 has four main components i.e. *Tutorials, Exercises, Strategies* and *Game rules*. It consists of many scenes for user interaction so that the design constraints discussed in previous section can be fulfilled. Strategies and game rules are described with text explanation in corresponding scenes. Tutorials consists of videos for teaching carrom skills and practice exercises for them. Complex exercises are also provided for testing user's skills. Detail architecture of tutor is described in dissertation of Mrinal Malick [10]. Sequence of activities when user interacts with tutor, activity diagram and various functions of tutor are also discussed in it [10].

Chapter 6

Implementation

Chapter 5 defines architecture, components and other characteristics of the *Carrom Tutor 2.0*. In order to provide desired features and functions discussed in chapter 5, a game like environment must be provided to users. **3D** environment can make it same as the real carrom playing experience. This kind of environment can be created, only with the help of any game engine. Blender is very popular and open source game engine used for creating games, animations, object models etc [7]. Blender Game Engine uses python scripts to control the behavior of objects in game and logic editor is used for assigning controllers to objects. So, it was decided to use *Blender Game Engine* for building *Carrom Tutor 2.0*.

In Blender, implementation of any functionality is done by using *python scripts* and *logic editing* together. These are applied on object models created in game. User interface of Blender Game Engine for Game Logic showing text editor of python scripts, logic editor, object outliner and 3D view of game is displayed in figure 6.1. Object modelling, python scripting and logic editing are three main pillars of implementation in blender. These are discussed in following sections.

6.1 Modelling

Objects, characters and scenes in the blender games are created by modelling. Blender provides very simple and convenient interface for modelling. Carrom board surface and side frames were created by cubic mesh objects. Coins and striker were made from cylin-

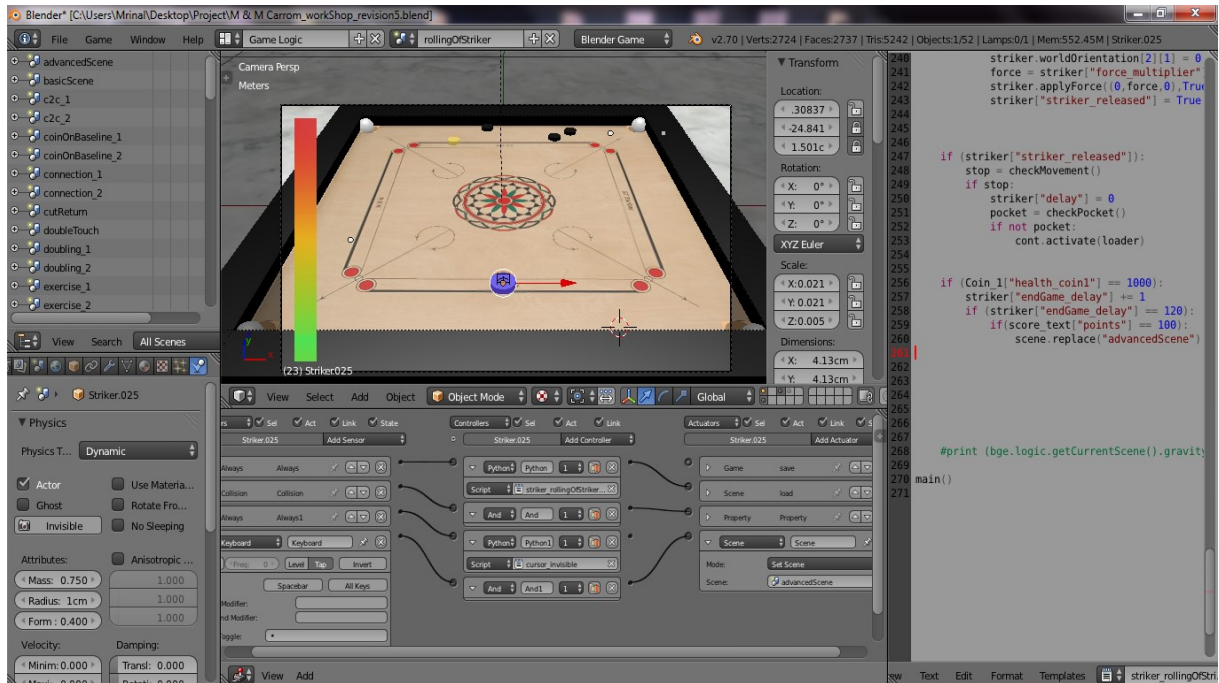


Figure 6.1: User Interface of Blender Game Engine

ders. Texture images were mapped on some objects in game. The room was created by using plane objects and some texture images were mapped on them. Force gauge and text objects for displaying score were attached with camera so that they move along with camera. Some more details about modelling can be found in chapter 9.

Animations of carrom skills were created in *Blender Render* by using timeline feature. States of objects were fixed for each keyframe on timeline. Sequence of these keyframes produces the animation. Output of these animations was taken as image sequences. Videos used in Carrom Tutor 2.0 for teaching carrom skills were created by using these image sequences.

6.2 Logic Editor

Behavior of objects in blender game can be controlled by setting up the logical components of that object. Blender Game Engine has user friendly graphical interface called *Logic Editor* for setting up the logical components of objects. In logic editor, there are three main components i.e. *Sensors*, *Controllers* and *Actuators*. Specific conditions or properties in game environment are sensed by sensors and according to set condition they

are triggered. Once a particular event occurs, sensor is triggered and it sends out a positive pulse. Different types of controllers are provided in blender for deciding the object's behavior. Output of sensors is given to controllers and controllers decide the action to be taken. Python scripts can be used as controller or output of sensors is used for triggering any actuator by controllers. Actuators are used for performing specific actions in game. Different types of actuators are provided in blender for handling game, scenes, objects, object properties etc. Many complex functionalities can be achieved by making proper use of sensors, controllers and actuators. Some important modules in Carrom Tutor 2.0 were build in logic editor. They are described below.

1. Dependencies between intra-Scene objects

For some complex functionalities, properties of an object were needed to be accessed by other object. This was done in *Logic Editor*. If multiple objects are selected together then *Logic Editor* opens them in a single layer and connections between logical components of different objects can be established. Consider *intermediateScene* as an example. There are five intermediate level skills and we have only one plane i.e *screenIntermediate* to show the videos of these skills to user. So these six game objects should be inter connected. Five "Python" controllers were added to object *screenIntermediate*. Each of these controllers is connected to "Property" sensor of one intermediate level skill. Similarly there are many such inter object connections in different scenes. Figure 6.2 shows intra-scene object dependencies in 'intermediateScene'.

2. Displaying videos and presenting practice exercises

Video of selected skill is displayed to user on *Screen* object. In *intermediateScene*, there is plane object named 'intScreen'. It has two properties i.e. global variables to ensure proper display of video. One is string variable named 'material' for storing the name of the screen object i.e *intScreen* and second variable is of type boolean and it is set to *False* so as to play the video only one time. In addition there are two variables for each intermediate level skill. For example, *Connection* skill is assigned string variable 'movie5' for storing the name of related video to be played and boolean variable named 'video_played5'. All these global variables are shown in figure 6.3.

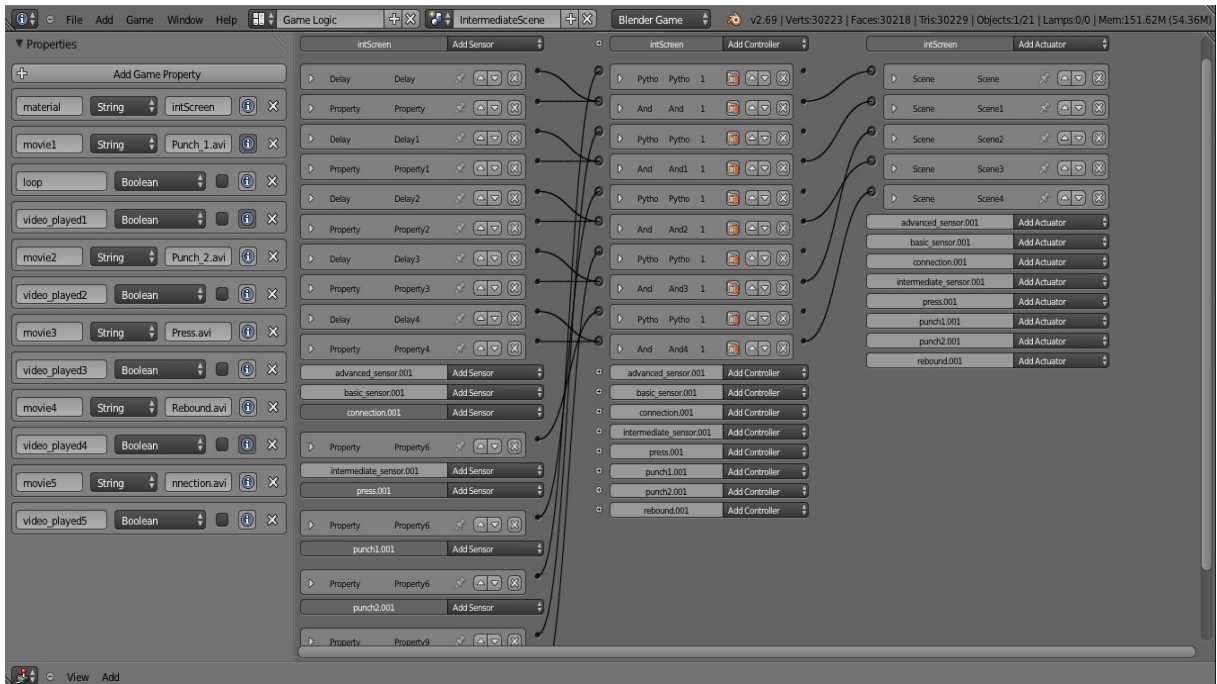


Figure 6.2: Intra-scene object dependencies



Figure 6.3: Global variables used in intermediate scene

In this case, name of the video is *Connection.avi*. There are two sensors in logic editor for each skill in intermediate level, delay sensor and a property sensor. ‘Connection’ skill has delay sensor to wait for specific amount of delay set in it and a property sensor applied on ‘video_played5’ variable to check whether its value is *True* or not. Both sensors are connected to ‘And’ controller. When both sensors send

a positive pulse, this controller triggers the ‘Scene’ type actuator connected to it. Scene named ‘connection.1’ is loaded by this actuator where first practice exercise is provided to user for playing. After successful completion of first practice exercise second practice exercise is loaded in scene ‘connection.2’. When user finishes both practice exercises, she is redirected to intermediateScene. Sensor, controllers and actuators used for playing video and loading practice exercises are shown in figure 6.4. Logic bricks for basicScene and advancedScene are formed in same way.

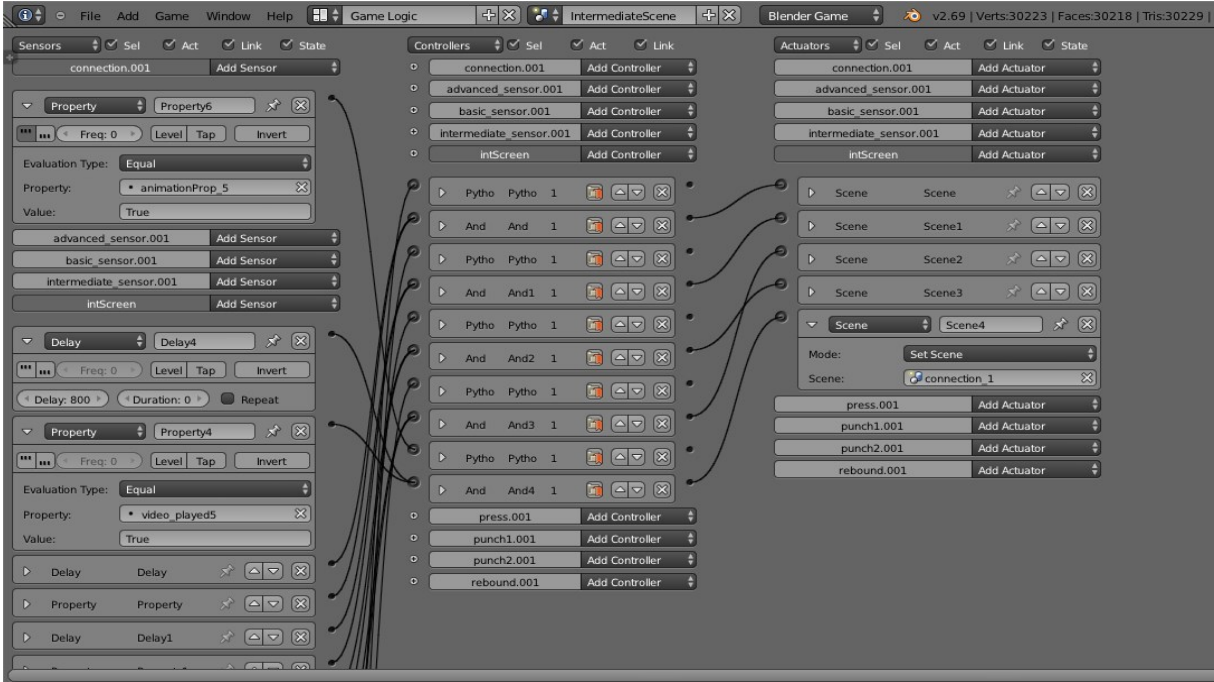


Figure 6.4: Logic bricks for playing video and loading practice exercise

Logical components for **Designing UI** and **Designing Exercises** are given in dissertation of Mrinal Malick[10].

6.3 Python Scripting

Python is general purpose scripting language and it has special interfaces to access all Blender’s internal functions from the language. It is interpreted, interactive and object-oriented programming language. Python scripts are written to extend the functionality of Blender. In order to interact with Blender, scripts can make use of the tightly integrated API (Application Programming Interface). This API can edit any data the user interface

can like scenes, meshes etc. and many such things.

Blender provides built-in Text editor and Python console for writing and executing the python scripts. We can open .py files in text editor and test them using the Run Script command. The Python Console is typically used for typing in snippets and for testing to get immediate feedback. We can write scripts to control the scenes, objects from scenes and perform many operations in the scene. Python scripts(.py files) are used as Controller in Logic Editor for controlling actions of objects in scenes of the game. These controllers are getting inputs from the sensors which are continuously sensing the game environment. We can choose the type of sensor for getting inputs and connect it to the controller. We can also initiate some actions provided by blender game actuators in scripts by connecting script controllers to actuators.

In *Carrom Tutor 2.0*, scripts were applied on striker, coins, empty objects and text objects in every scene for achieving the required functionality of tutor. In order to obtain the necessary features many python scripts were written and some were downloaded from blender community. Following are some main parts of the scripting in *Carrom Tutor 2.0*.

1. *Making User's view according to mouse movements*

As discussed in chapter 4, it has been decided to make tutor more user friendly. Users should be able to play actual shots and see results instead of just watching the GIFs for their selected shots. It has been decided to provide the *3D view* to users for playing shots just like first person view in games to give them feel of real carrom board scene.

For this, camera is situated at the position where player sits for playing carrom and it is given an angle like a person watching to board. The mouse movements are captured and applied to the camera by the script named "*mousemove.py*". Mouse movements are sensed by this script and then accordingly the user's view is changed like first person view in games. Camera is made the child of empty object named "Empty" which is placed with striker and "*mousemove.py*" is applied on this empty object. This script is available on blender community [13]. Output of 'Mouse' movement *sensor* is given to the controller, which will be the script "*mousemove.py*".

2. *Showing striker's path as a line*

A red line showing the striker's path is drawn in the scene for giving users a good idea about striker's movement after release. As the mouse is rotated by user, the view gets rotated and according to that direction the line gets rotated to show striker's path. This line is drawn dynamically in the script named "**RaySensor.py**". One empty object named "**RayEmpty**" is placed with the striker and it is allowed to move with striker. Sensor called Ray is applied on this object which throws a ray in positive Y axis direction of empty object. And then the line is drawn from position of RayEmpty object to the point where the ray hits some other object by the controller i.e. "RaySensor.py".

3. *Striker's orientation and applying force*

A python script "**striker.py**" is handling most important part of the game including force gauge, applying force on striker in proper direction, changing between the scenes after some condition, moving striker on base line etc. This script is connected to **Always** sensor for taking inputs from the game environment. This Always sensor gives continuous positive pulse to its controller. "striker.py" runs sixty times in a second and all the variables in it get values in all executions. So, some global variables were required to store game features. Many properties of type boolean, integer and float were created for object 'Striker' to serve the purpose of global variables in scripts.

Left and right arrow keys are used to position the striker on baseline to play desired shot. The striker placing is restricted within two red balls for playing and users are not allowed to play with striker placing on half ball. Python code snippet implementing these conditions is shown below.

Here lKey, rKey and lClick variables are set as left arrow, right arrow and left click buttons are pressed respectively. Once user presses the left click then force gauge will get enabled on screen which is continuously advancing up and down. This slider represents the force amount for playing the shot. When user clicks for second time, amount of force proportional to the part of slider visible to user at that time is applied on the striker in its **Y axis** direction.

```

lKey = bge.logic.KX_INPUT_ACTIVE ==
    keyboard.events[bge.events.LEFTARROWKEY]
rKey = bge.logic.KX_INPUT_ACTIVE ==
    keyboard.events[bge.events.RIGHTARROWKEY]

lClick = bge.logic.KX_INPUT_JUST_ACTIVATED ==
    mouse.events[bge.events.LEFTMOUSE]

    if not striker["slider_enabled"]:
        if (lKey and striker.position[0] > -21.62116):
            striker.applyMovement((-0.1,0,0),False)
        if (rKey and striker.position[0] < 22.45835):
            striker.applyMovement((0.1,0,0),False)
        if lClick:
            if((striker.position[0] <= -18.31307 and
striker.position[0] >= -20.75888) or (striker.position[0] <=
21.53248 and striker.position[0] >= 18.84937)):
                print("Wrong striker placing")
            else:
                striker["slider_enabled"] = True
                lClick = 0

```

While playing the shot, red line showing striker's path rotates according to mouse movements. This is done by using the RayEmpty object, but rotating the striker in that direction is very difficult. We need to rotate the striker same as the camera so that striker will move in the direction where user is looking. Camera is rotating around all axis, but striker should not rotate around X and Y axis. It must rotate around Z axis only and its X and Y axis directions should change. Python API for blender provides a method called *“orientation”* which gives orientation of object in 3D world. Orientation of every object is stored in 3×3 matrix. We need only X and Y axis orientation of camera for striker. After the second click, striker is released and it will move in user's desired direction. X and Y axis orientations of camera are applied to striker before releasing it so that it will move in direction where user is aiming. Part of the code from “striker.py” executing this is shown below.

Initially, “force” variable is set to zero and it is assigned a value according force slider after second click. Force of this magnitude is applied to striker in its Y axis direction.

```

force = 0
    if not striker["striker_released"] and lClick:
        vector_magnitude_x = math.sqrt((camera.worldOrientation[0][0]
            ** 2) + (camera.worldOrientation[1][0] ** 2))

        vector_magnitude_y = math.sqrt((camera.worldOrientation[0][1]
            ** 2) + (camera.worldOrientation[1][1] ** 2))

        striker.worldOrientation[0][0] = camera.worldOrientation[0][0]
            /vector_magnitude_x

        striker.worldOrientation[1][0] = camera.worldOrientation[1][0]
            /vector_magnitude_x

        striker.worldOrientation[2][0] = 0

        striker.worldOrientation[0][1] = camera.worldOrientation[0][1]
            /vector_magnitude_y

        striker.worldOrientation[1][1] = camera.worldOrientation[1][1]
            /vector_magnitude_y

        striker.worldOrientation[2][1] = 0
        force = striker["force_multiplier"] * 90
        striker.applyForce((0, force, 0), True)
        striker["striker_released"] = True

```

Python code for *Displaying videos*, *Force gauge* and *Scoring in exercises* is explained in thesis written by Mrinal Malick [10]. Python script for displaying videos was downloaded from online community of blender artists [8]. Using these scripts as controller and their settings in game environment were given on blender artist community [5].

Chapter 7

Demonstration

This chapter includes details of demonstration of *Carrom Tutor 2.0*. Start screen of this tutor has following three buttons as shown in figure 7.1.



Figure 7.1: Start screen

- **Start** : It opens the second screen containing another four options
- **How to** : Instructions about how to play in exercises are displayed on same screen when user clicks on this button.

- **Quit** : This button is used to exit from tutor.

The second screen contains four buttons which are links to the major functionalities in tutor. Those buttons are :

- **Tutorial** : It directs to the screen where user can find demos of carrom skills and practice exercises.
- **Exercise** : It is a link to complex exercises which users are supposed to complete after going through tutorials.
- **Strategies** : This button loads screen containing textual explanation of various strategies used in carrom game.
- **Game rules** : List of carrom game rules is displayed to user.

Figure 7.2 shows second screen of the Carrom Tutor 2.0.



Figure 7.2: Second screen

The tutorial scene contains three buttons viz. Basic, Intermediate and Advanced. Various carrom skills discussed in chapter 2 are divided in these three categories. When

user clicks on any of these buttons corresponding list of skills in displayed. If user further clicks on any skill, video explaining that particular skill is presented to user on right half of the screen. Tutorial scene with list of intermediate skills and a video played on right half of screen is shown in figure 7.3. After watching video, user is provided two practice exercises for implementing same skill in it. If user plays it wrong, same exercise scene gets reloaded.

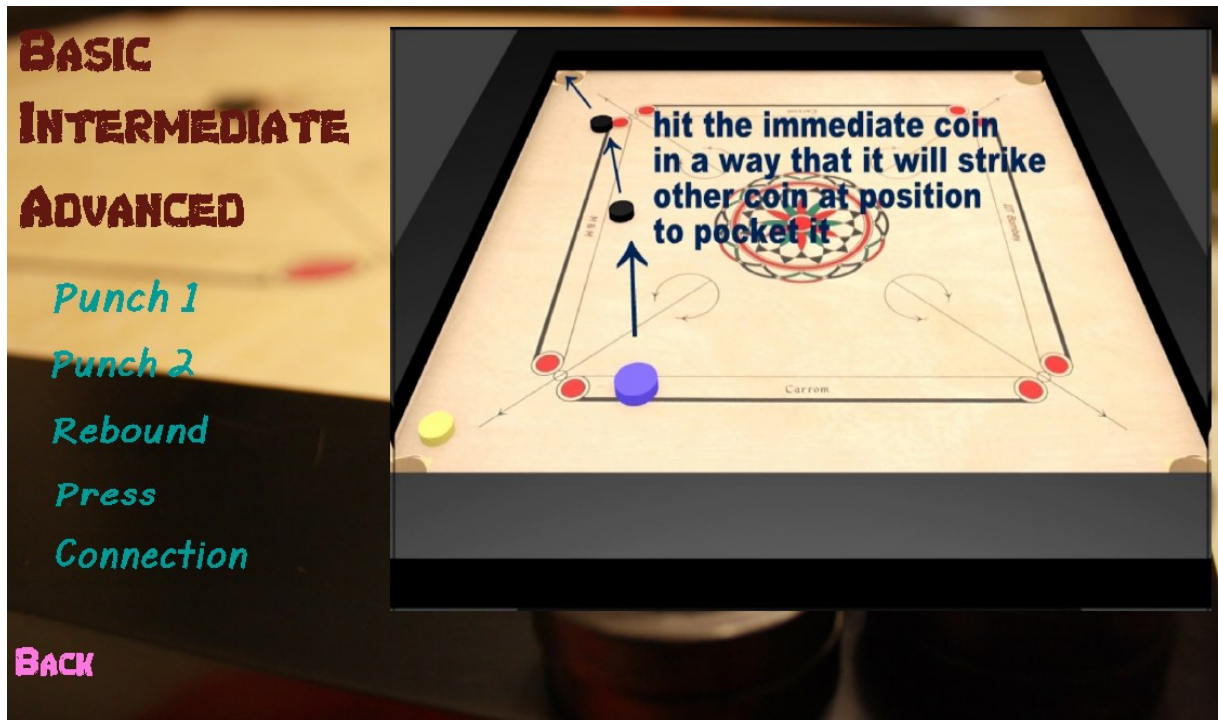


Figure 7.3: Tutorial screen

Complex exercises are provided to users in exercise scene. Corresponding exercises are loaded after clicking on the button. In each exercise, a complex board situation is given to users for applying their skills. User must pocket all black coins without giving a chance to opponent to win the game and get maximum points. For each shot played by user, some points are given. User's score in exercise is displayed in right upper corner of the screen. If user fails to pocket all coins, then same exercise scene is reloaded. Figure 7.4 shows the third exercise scene in Carrom Tutor 2.0.

Strategies used in carrom board games are listed in the scene "Strategies" with textual explanation. Also carrom game rules are given in scene "Game Rules". Second scene shown in figure 7.2 contains the links for these scenes.



Figure 7.4: Complex exercise

Chapter 8

User Experiments

This chapter discusses about the methodology followed for conducting the user experiments of *Carrom Tutor 2.0*. Target audience, the way in which data is collected and analysis of collected data is described in following sections.

8.1 Sample

Learning objective of the Tutor is that the user should be able to identify the Carrom skills and apply some of those skills in a given board situation. So the target audience of the tutor is required to know the Carrom board game and played it some times. Data of the user experiment is collected from these samples. Eleven users took part in the user experiment of *Carrom Tutor 2.0*. This group of users was mixture of beginners, intermediates and experts in Carrom.

8.2 Data Collection Methodology

We decided to conduct pre-post tests for data collection. In addition, *Carrom Tutor 2.0* is compared with other carrom applications based on end user experience. Complex exercises are provided to users before going through tutorials and practice exercises. After this, users were asked to go through all tutorials and practice exercises step by step and then they played complex exercises again. This was achieved purposefully, so that we can check level of difficulty experienced by user while playing exercises before and after

using tutor. Each user was given an online carrom application, *Carrom King* for playing [9]. This application is similar to other carrom games available online. In feedback form, users were asked to compare *Carrom Tutor 2.0* with *Carrom King*. Half of the users played *Carrom King* before interacting with *Carrom Tutor 2.0* and remaining users did it in opposite manner by playing *Carrom Tutor 2.0* first. Users were asked to give their feedback in a form. Questions asked in that form and more details about the data collection methodology can be found in dissertation written by Mrinal Malick [10].

8.3 Data Analysis and Results

For *Carrom Tutor 2.0*, users were requested to fill the *SUS* form so that we can get the feedback about the accessibility, efficiency, effectiveness and attractiveness of this system [1]. All users either “agreed” or “strongly agreed” to use the system frequently in future. When asked about whether the system is complex to use, everyone either “strongly disagreed” or “disagreed”. Question 3 asks the simplicity of the system. One user gave the comment “menus are self explanatory”. The responses were either “agree” or “strongly agree”. All users either “strongly disagreed” or “disagreed” when asked if assistance of a technical person is required in question 4. In question 5 it was asked whether various functions of this systems were well integrated or not. Responses we got were mostly “agree” or “strongly agree”. Only one user kept a “neutral” opinion about this question. All of the users “strongly disagreed” or “disagreed” about the question whether the system has too many inconsistencies. Every user “strongly agreed” or “agreed” upon the fact that one can learn to use the system very quickly. Most of the users praised about the user interface design. “strongly disagreement” or “disagreement” received as responses when asked if the system is very complex to use. Some user responded “it was fun” to interact with the Carrom Tutor 2.0. Users “strongly agreed” or “agreed” about being confident while using the system. Some comments like “because it was easy to use, and interface was nicely built” was given as a reason. Most of the responses were “strongly disagree” when questioned if they need some prior training to use this system. Some useful suggestions were also received. One user had little difficulty to release the striker at the desirable force, so the suggestion was to reduce the up-down speed of slider a little. Some users told to add replay option for videos and put a help option in complex exercises.

Average *SUS* score of *carrom Tutor 1.0* was **77.14%**. It falls in grade ‘B’ systems for its usability according to its score [15]. Average percentage of *SUS* score for *Carrom*

Tutor 2.0 is 84.09%. As its score is more than 83% it is a grade ‘A’ system [15]. Average responses given to each question asked in SUS analysis have been plotted in figure 8.1. According to the average SUS percentage, Carrom Tutor 2.0 has surpassed its predecessor. It can be inferred that user flexibility, interface design, content design play important roles in a tutor design.



Figure 8.1: SUS Feedback for tutor

Questions asked for measuring user’s learning gain and their analysis have been described in thesis written by Mrinal Malick [10]. Also user’s feedback and suggestions about *Carrom Tutor 2.0* are mentioned in it.

Chapter 9

Challenges

Tutor must provide more flexibility and control to users as discussed in chapter 4. To make it like a real experience of playing, user should be able to see her played shots. This can be done in a game like environment. *Blender Game Engine* is used to provide more flexible environment. Learning various functions and utilities of game engine was challenging and it took some time to understand. First step towards designing the tutor was to create a carrom board, coins and striker. Board surface and outer frames were created from cubical mesh objects. Coins and striker were created from cylindrical mesh objects with exact dimensions. In order to increase the smoothness of pocketing a coin some object modelling were done. It was needed to pocket coins similar to what happens in real life carrom game. Four invisible spheres were placed inside four pockets. These spheres detect collision with coin objects. If collision is detected, it reduces coin's Z axis coordinate value by 2 units. There are small planes placed under each pocket. Collision of coins with these planes identified to decide whether the coin is pocketed or not. It was very challenging to implement coin pocketing by using these objects. In addition, two empty objects were used in exercises. First object was created for drawing a line for showing striker's path and second object is a parent of camera for making the rotation and movement of user's view similar to the real experience. Using the empty objects in this way was somewhat tricky.

It was very important to understand the use of python script as controller of any object and how to access other objects in python script. It needs the knowledge of python API for blender and library functions available. It took some time to get used to this and understand its functions. In exercises, camera's orientation is applied to striker before

releasing it from baseline. This is achieved by using “orientation” method provided in python API for blender. Orientation of object in 3D world is stored in 3×3 matrix. Each column of this matrix stores the orientation of one axis in 3D world. Column 0,1 and 2 store orientation of X, Y and Z axis respectively. Identifying this format of storage and applying specific axis orientations to striker was very time consuming. Some vector calculations were performed with orientation vectors of axes to apply force on striker in correct amount.

Left and right arrow keys are used by user to place striker on desired position in game scene. For playing shot, force gauge is enabled with one left click. Force gauge is continuously moving up and down between lowest and highest values for force. This up and down movement of slider is achieved by making ten “Slider” planes visible and invisible as discussed in chapter6. Movement of force gauge is made slow near lowest and highest points in order to make it easy for users to catch these peaks. For this, sine wave characteristics were applied to slider movement with the help of “timer” variable. Depending on the value of timer variable “Slider” planes are made visible and invisible. Setting the visibility of planes according to varying values of timer variable was challenging.

As discussed in chapter6, two available scripts on blender community were used for capturing the mouse movements and displaying the videos of carrom skills to user. Understanding these two scripts and applying them in Carrom Tutor 2.0 was crucial.

Different exercises were created in different *.blend* files at the beginning. These files can be called in other *.blend* file. But if the called *.blend* file terminates then control will not come back to calling file. So, different exercises were created in different scenes of same *.blend* file to easily transfer the control.

Chapter 10

Conclusion and Future Work

10.1 Conclusion

We have built a tutoring environment for teaching carrom skills and strategies to carrom aspirants. In this report, we describe carrom skills along with novel strategies and effective methods for teaching carrom. A web based system, *Carrom Tutor 1.0* was produced for teaching carrom skills and strategies. It also provides exercises to users for testing their skills. User experiments were conducted on Carrom Tutor 1.0 for checking the usability of this system and for measuring the learning gain in users. It is a grade ‘B’ system according to SUS (System Usability Scale) analysis for usability. Many improvements required in the tutor were suggested by users in experiments. Carrom Tutor 2.0 was built using *Blender Game Engine* in order to provide more flexibility and control to users in game. Opportunities for learning carrom skills and strategies are provided to users in 3D game environment. User experiments were conducted on Carrom Tutor 2.0 and it was compared with other carrom applications also. User experiments shows that, Carrom Tutor 2.0 is a grade ‘A’ system with more interactivity. It can be seen that performance of Carrom Tutor 2.0 is better than Carrom Tutor 1.0 as its usability is higher. It is a more attractive, efficient and effective system for teaching Carrom skills and strategies.

10.2 Future Work

Game based carrom tutor provides less number of complex exercises to users for testing their carrom skills and strategies. More number of exercises can be added to improve the

control of user on force and aiming in carrom game.

A full carrom board game can be provided to users for testing their skills by playing with opponent. An artificial intelligence can be added to play game against the user. This will provide a good environment to users for applying their learnt skills and strategies.

Blender game engine used for building the tutor sometimes give different results for the same condition provided. There was a problem with collision bounds of objects in scene. This causes unexpected behavior of objects colliding with each other. If these problems are solved, tutor experience will become more effective.

Bibliography

- [1] SUS analysis application. <http://www.mohini.0fees.net/iit/welcome.php>.
- [2] U S Carrom Association. <http://www.carrom.org/>.
- [3] John Brooke. *SUS - A quick and dirty usability scale*. Digital Equipment Corporation, 1986.
- [4] Allan Collins. *Cognitive Apprenticeship*, chapter 4, Handbook of the Learning Sciences. Cambridge Univ. Press, 2006.
- [5] Blender Community. <http://www.blenderartists.org/>.
- [6] Paul J. Diefenbach. Practical game design and development pedagogy. *Published by IEEE Computer Society*, pages 84-88, May/June 2011.
- [7] Blender Game Engine. <http://www.blender.org/>.
- [8] Tutorials for Blender 3D. <http://www.tutorialsforblender3d.com>.
- [9] Carrom King. http://twoplayergames.org/play/719-Carrom_King.html.
- [10] Mrinal Chandra Malick. Carrom tutor : Game-based learning and implementation. Master's thesis, IIT Bombay, June 2014.
- [11] Mr.Somanchi. Carrom play techniques. <http://uscarrom.org/docs/CarromPlayTechniques.pdf>.
- [12] Tom O'Donnell. *Chess Teaching Manual*. Chess Federation of Canada, 1997.
- [13] Riyuzakistan. 3d art - game design. <http://www.riyuzakistan.weebly.com>.
- [14] Java server pages. http://en.wikipedia.org/wiki/Java_server_pages.
- [15] Measuring usability. <http://measuringusability.com/sus.php>.
- [16] Wikipedia. <http://en.wikipedia.org>.