Documentation

Carrom

- **Major Parts of the Game**
  1. Designing the board display
  2. Taking player input
  3. Calculating game state and motion of pieces
  4. Sending and recieving data over the network
  5. Co-ordinating game flow
  6. Creating computer players
  7. Testing and debugging

- **Structures used**
  1. colour - float r, float g, float b
     The three indices for colouring
  2. CIRCLE - colour c, float r, float x, float y, float vx, float vy, int in
     Attributes of a circle ie colour, velocity, and cooridinates
  3. BAR - int x, int y
     Co-ordinate structure
  4. sides - char ip[20], int there, int sock
  5. game - sides side[4], float friction, float erest, int playercount, int turn, int me
     All the information about a game's state needed to start/join

- **Functions used**
  Format - return_type function([data_type parameter])

- Drawing
  handleResize(int w, int h) - Manages a window resize operation
  createCircle (float r, float h, float k, float width)
- Game Physics
  - int circ$_{e}q(CIRCLEa, CIRCLEb) - 1 if a and b are same, else 0$
  - void moveCircle (float r, float* h, float* k, float width, float* vx, float* vy, float red, float green, float blue)
  - int hasstop() - 1 if every piece has stopped, 0 otherwise
  - void col(CIRCLE *a,CIRCLE *b) - collide two CIRCLE structures
  - void ispot(CIRCLE *git) - Prints where the CIRCLE 'git' got pot
  - int checkcol(CIRCLE a,CIRCLE b) - 1 if a, b collide, else 0
  - int getfrictioncoef(void)
- Networking
  - int startserver(int port) - 1 if working, 0 if not
  -void error(const char *msg) - exits
  - int startclient(char *ipaddr, int port) - 1 if working, 0 if not
  - void decrypt(void) - Decrypts network message
  - void createmsg(int type,int button,int state,int power,int x, int y)
  - void *check4msg(void *argv)
- Input
  - void handleKeypress(unsigned char key, int x, int y) - Keyboard Operations like rotation, power level and exit
  - void handleMotion(int x, int y1)
  - void handleMouse(int,int,int, int)
  - void drawScene(void)
- Initialize
  - void Init_var(void)
  - void initRendering(void)
  - void initgameinfo(void) - Initial State of the system
- Manage Game state
  - int checkpos(char *buffer,int *vicside,char *ipaddress)
  - void sendgameinfo(char *buffer)
  - void *mang_all(void *portno)
  - void *mang_alls(void *argv)

- **Common User Interface**
  1. Carrom Board
     This includes properties like structure of board, friction, etc.

2. Pieces

   The number and absolute arrangement and motion of pieces has to be the same for everyone

3. Rules

   All the rules are absolute and have to be same for game to be viable.

   (a) To play, striker should be struck from bars on the board
   (b)
   (c) Players should pot only the assigned colour coins
   (d) Potting a right coin results in another turn, else next player gets to hit
   (e) Queen can only be won if another coin follows q
   (f) The covering coin decides the owner of the coin
   (g) In a 4-player game, opposite players are partners
   (h) If hit hard, coin may jump out of the board
   (i) First one to finish his coins on the board wins

- **Player Dependant States**

  1. Board Display

     How board looks to each player. This also include personalization using rotation of the board.

  2. Gameplay

     Every player should be able to control motion of striker (power, direction, location) only on his turn.

  3. Control Options

     User should be able to change control keys (and devices) according to his preference.

- **Network Communications**

  – Each machine connnected with every othermachine thus enabling direct communication(chat).
  – New Player can join through any machine and at any time.
  – On joining the game state(basically contents of game info) passed onto the player.

  1. Local Operations - Not Transmitted
     - Board Display and Rotation
     - Strike Power and Direction
     - Personalized messages (from game to a specific player)
     - Physics of collisions

2. Network Operations - Transmitted to/from server and Broadcasted to every player
   - Boards State (Striker and Pieces)
   - Parameters for motion of pieces
   - Data (like Player Name, Score, etc)

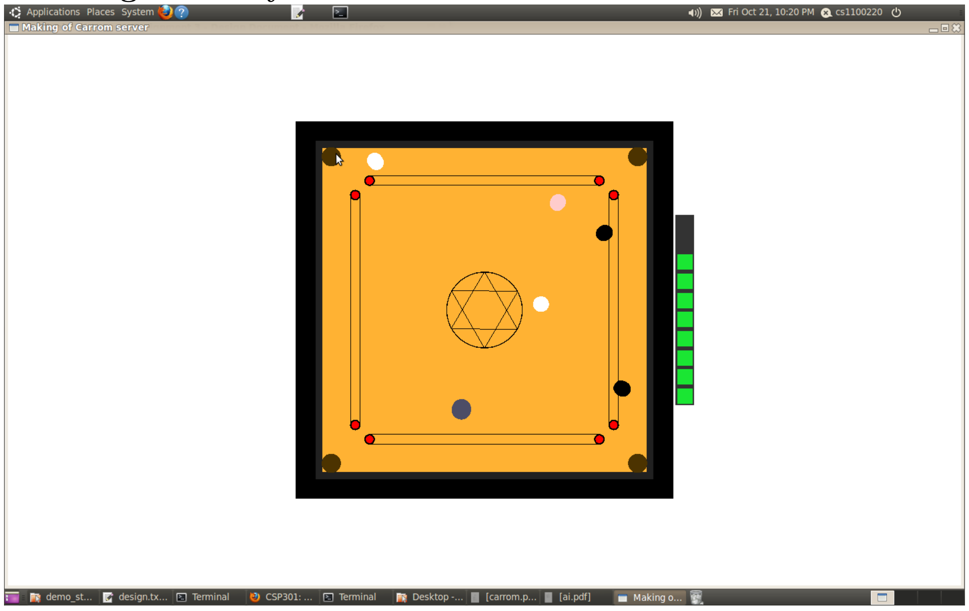
- **AI**
  - Depth calculation -
    Calculates possible future game state with level of calculations varing from no future collision taken into account while aiming to pot coin to 3-4 level (no. of collisions) taken into account
  - Difficulty of shot based upon:-
    Angle - Straight line or complex shot
    Coin involvement - May pot a coin using another coin
    Reverse Shots
    Confidence Zones - The preferred parts of the board
  - Implementation:-
    - Varying apparent pot hole size for different levels
    - Variation in the accurate angle
    - Low level bot just decides power on distance, while higher level take into account the collision
    - Higher level bot can line up other coins specially for queen.
    - Destruct other's possible favourable positions
    - Confidence zones describe the regions which bot prefers, and probability of shot depends on it

- **Threads (one for each of the following)**
  1. Personalized thread for each player :
     if game starter - waits for the remaining players to connect and sends them the game info if game joiner - connects to existing players, receives game info, starts a new thread which makes it now act as a server
  2. Physics thread : Will help perform independent calculations parallely and thus reduce computation time.
  3. Data receiving threads: These wait for messages from other machines and interprets the message received, For each connected machine there is a different thread to handle communication.

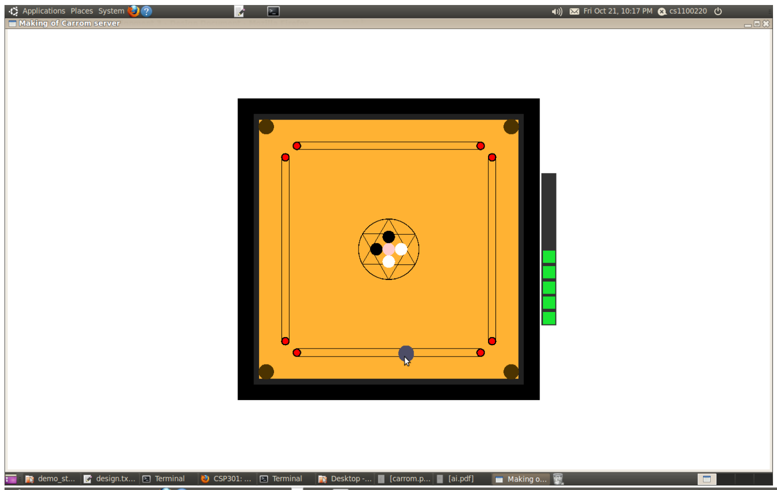
- **Gameplay Defaults**
  1. Rotation along X-axis a-d

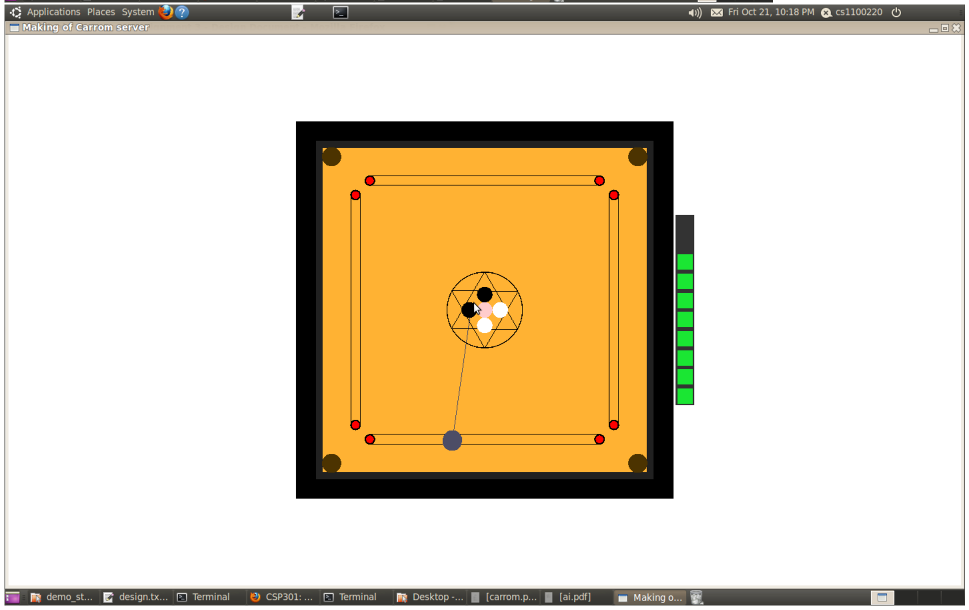
2. Rotation along Y-axis w-s
3. Rotation along Z-axis z-x
4. Striker options
   - Press Enter key to decided Power level
   - Move mouse to fix location on bar Left Click to Select
   - Move mouse to fix direction of shot Left Click to Select
5. Run Options - ./carrom
   Parameters to be specified while doing make carrom
   - Play Mode:-
     1 -
     2 - client
     3 - offline
   - Friction level :- 1-10
   - Coefficient of Restitution with Walls - e

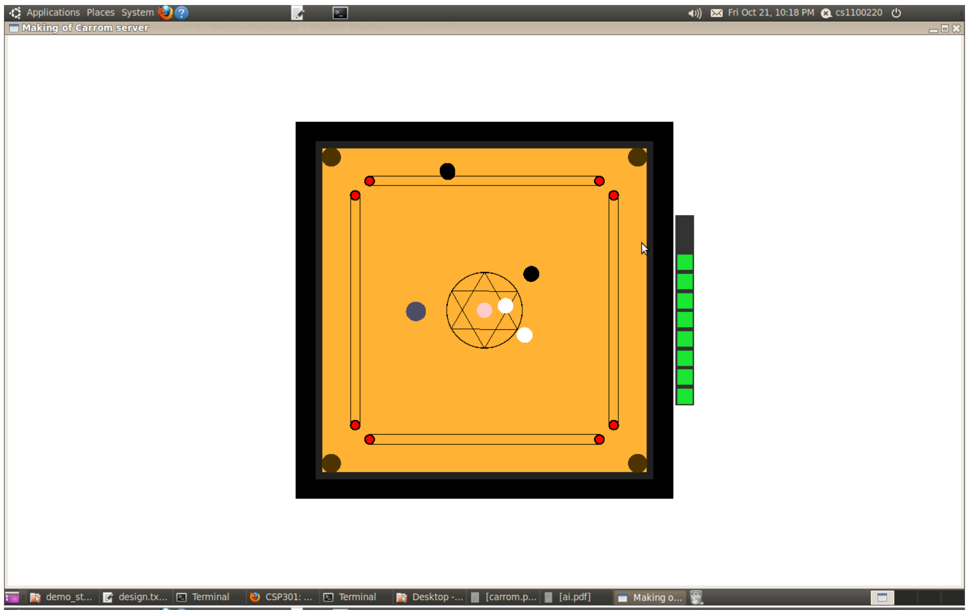
- **Working of the system as a whole**

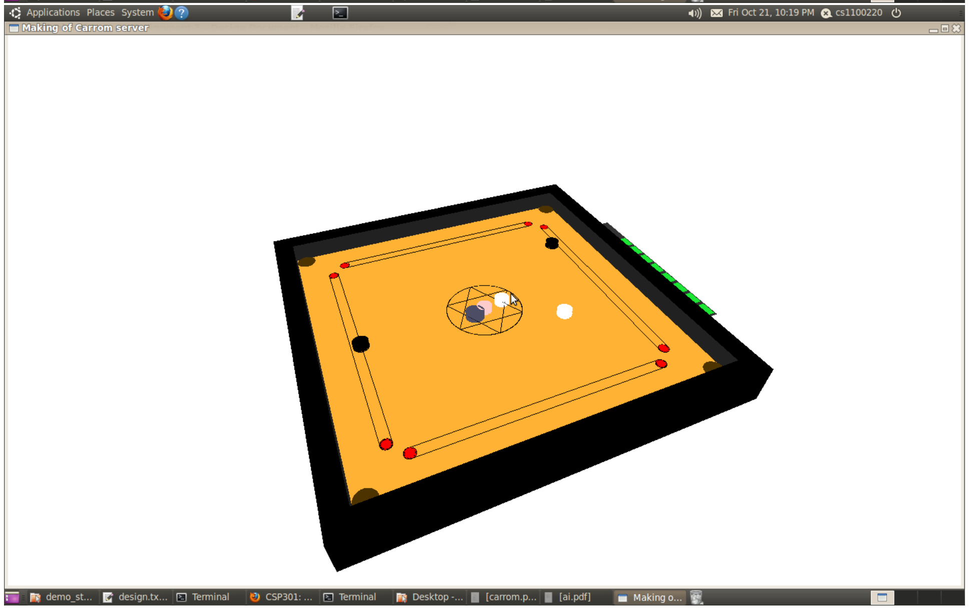


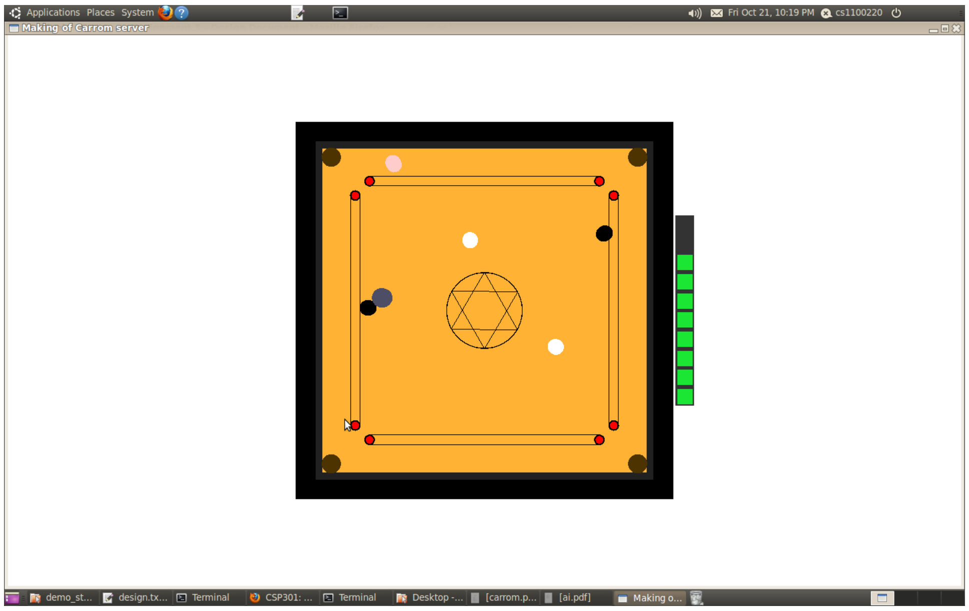
- **Screenshots**


Applications Places System
Fri Oct 21, 10:17 PM
cs1100220
Making of Carrom server
demo_st...
design.tx...
Terminal
CSP301: ...
Terminal
Desktop -...
[carrom.p...
[ai.pdf]
Making o...


Applications Places System
Fri Oct 21, 10:18 PM
cs1100220
Making of Carrom server
demo_st...
design.tx...
Terminal
CSP301: ...
Terminal
Desktop -...
[carrom.p...
[ai.pdf]
Making o...


Applications Places System
Fri Oct 21, 10:18 PM
cs1100220
Making of Carrom server
demo_st...
design.tx...
Terminal
CSP301: ...
Terminal
Desktop -...
[carrom.p...
[ai.pdf]
Making o...


Applications Places System
Fri Oct 21, 10:19 PM
cs1100220
Making of Carrom server
demo_st...
design.tx...
Terminal
CSP301: ...
Terminal
Desktop -...
[carrom.p...
[ai.pdf]
Making o...


Applications Places System
Fri Oct 21, 10:19 PM
cs1100220
Making of Carrom server
demo_st...
design.tx...
Terminal
CSP301: ...
Terminal
Desktop -...
[carrom.p...
[ai.pdf]
Making o...