Engineering Logbook

Category

[1. Websites 2](#_Toc39681523)

[2. Literature 3](#_Toc39681524)

[a) Literature: 3](#_Toc39681525)

[b) Thesis: 3](#_Toc39681526)

[3. Notes 4](#_Toc39681527)

[(1) Summary Before 2020/04/07 4](#_Toc39681528)

[(2) 04.04: Server Programming 4](#_Toc39681529)

[(3) 04.05: Server Programming 4](#_Toc39681530)

[(4) 04.07: Meeting 4](#_Toc39681531)

[(5) 04.09: Bug Fix 4](#_Toc39681532)

[(6) 04.10: Learning 4](#_Toc39681533)

[(7) 04.14: Programming 5](#_Toc39681534)

[(8) 04.17: Learning & Bug Fix 5](#_Toc39681535)

[(9) 04.22: Programming 5](#_Toc39681536)

[(10) 04.24: Learning & Debug 5](#_Toc39681537)

[(11) 04.27: Programming 6](#_Toc39681538)

[(12) 04.30: SoBT meeting 6](#_Toc39681539)

1. Websites
   1. GitHub: <https://github.com/JYATMorz/BachelorThesis>
   2. GitKraken: <https://app.gitkraken.com/glo/board/XlUkZKMmogAQi2U6>
   3. Phaser3 API Document: <https://photonstorm.github.io/phaser3-docs/index.html>
   4. Phaser3 Examples: <http://phaser.io/examples>
   5. Socket.IO API Document:
      * <https://socket.io/docs/server-api/>
      * <https://socket.io/docs/client-api/>
   6. Tutorial (Chinese):
      * JavaScript: <https://www.runoob.com/js/js-tutorial.html>
      * Node.js: <https://www.runoob.com/nodejs/nodejs-tutorial.html>
      * HTML: <https://www.runoob.com/html/html-tutorial.html>
   7. More...
2. Literature
   1. Literature:
      1. <https://phaser.io/tutorials/making-your-first-phaser-3-game/part1>
      2. <https://gamedevacademy.org/create-a-basic-multiplayer-game-in-phaser-3-with-socket-io-part-1/?a=13>
      3. <http://buildnewgames.com/real-time-multiplayer/>
      4. <https://phasertutorials.com/how-to-create-a-phaser-3-mmorpg-part-1/?a=13>
      5. <https://phaser.discourse.group/t/phaser-3-real-time-multiplayer-game-with-physics/1739>
      6. <https://www.dynetisgames.com/2017/03/06/how-to-make-a-multiplayer-online-game-with-phaser-socket-io-and-node-js/>
      7. <https://gamedevacademy.org/phaser-3-tutorial/>
      8. <https://itnext.io/making-a-multiplayer-blockchain-game-using-phaser-nodejs-and-ethereum-pt-2-702b0c667e44>
      9. <https://medium.com/@tglaiel/how-to-make-your-game-run-at-60fps-24c61210fe75>
      10. <https://www.codementor.io/@amoslaber/build-a-paper-plane-with-phaser-3-and-arcade-physics-s8so8u831>
      11. More...
   2. Thesis:
      1. <https://scholar.google.com/scholar?start=30&q=distributed+systems+html5&hl=zh-CN&as_sdt=0,5>
      2. <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.452.1731&rep=rep1&type=pdf>
      3. <https://dl.acm.org/doi/abs/10.1145/2305484.2305527>
      4. <https://ieeexplore.ieee.org/abstract/document/6703321>
      5. <https://ieeexplore.ieee.org/abstract/document/6598536>
      6. <https://link.springer.com/article/10.1007/s11042-017-5242-4>
      7. <https://dl.acm.org/doi/abs/10.1145/2695664.2695987>
      8. More...
3. Notes
   1. Summary Before 2020/04/07

Waiting for sorting and editing

* 1. 04.04: Server Programming
     1. Introducing Socket.IO into the server
     2. Add socket.io events emitter and listeners to handle issues of synchronizing clients => The clients can perform same action at the same time now
     3. Turn shooting logic into a broadcast event with data checking on server
     4. Current Issues:
        + Server don’t know the exact order and all action is only fine when working on the first client
        + More than 4 users can connect which will cause errors
        + Random player position for each client due to built-in game logic => Ask first client to share its players’ position
        + No editable user names
  2. 04.05: Server Programming
     1. The server now can store the user data and delete them when users connect and disconnect. => using array and setting attributes to *null* or *splice* the array
     2. Using socket.io id as the unique id for each client and check id before every broadcast event (e.g. shooting ball)
     3. The 5th and later user will not connect to the existing game
     4. Current Issues:
        + Only the team with a client can be controlled => AI || single player
        + The 5th and later user will not being notice that they won’t start the game.
        + The later client’s action won’t be broadcasted to earlier client.
  3. 04.07: Meeting
     1. Add comment (the SHA of the commit) in cards of GitKraken when it’s done & Add description (the URL of the card) when committing updates in GitHub
     2. Upload and continue fulfill logbook (Catalog needed)
     3. Try to fill the content into the template of the thesis paper
     4. Local Network Test (ASAP)
     5. Next Meeting: 04/28 (Complete the Application with certain distributed systems architecture => Peer-To-Peer & Clients-Server; Basic test design)
  4. 04.09: Bug Fix
     1. The existing clients don’t receive the event of adding new user and deleting disconnected user. => Add new socket.io event, “*newUser*” & “*playerDown*”, to handle the stored socket id in existing clients.
  5. 04.10: Learning
     1. JavaScript ES6: Promise Object
        + <https://www.runoob.com/w3cnote/es6-promise.html> (Chinese)
        + Try to force new client to load latest position only when all players and ball are still. (The function keeps waiting until the require is met.) => Event Emitter won’t work due to asynchronous callback function.
  6. 04.14: Programming
     1. All clients won’t turn to next round until every client sends signal to the server.
     2. All clients won’t start game until every connected client sends signal to the server.
     3. Add simple AI logic at server side to control the teams that aren’t bonded to the clients.
  7. 04.17: Learning & Bug Fix
     1. Phaser 3 and JavaScript have different ways to calculate coordinate/angle.
        + JavaScript: Math.atan(x) -> (0 is right and 90 is up) or Math.atan2(y, x) -> (0 is right and 90 is down)
        + Phaser 3: GameObject.angle/rotation -> (0 is right and 90 is down)
     2. AI now can hit the ball correctly according to the positions of ball, player and door. It will go back if the chosen player is between the ball and opponent’s door, or it will try to shoot ball if the ball is between the chosen player and opponent’s door.
  8. 04.22: Programming
     1. The game now supports touch control.
     2. Add goal logic into the game.
     3. A more detailed playground is added.
     4. Detailed GUI with notification, title and score.
  9. 04.24: Learning & Debug
     1. The arcade physics of Phaser 3 uses static delta refresh time, the FPS can be adjusted by configuring ‘fps’ in ‘game config’. Different fps may cause different physical effect on different devices, especially the circular body, due to GPU or CPU performance. Forcing a static fps can greatly reduce the existence of different behaviors.
     2. A third-party API, ‘socket.io-p2p’, allows using the grammar of ‘socket.io’ to implement p2p connection between clients.
        + <https://github.com/socketio/socket.io-p2p>
     3. The FPS is set to 25 for better performances on both mobile and PC browsers.
     4. The game will check the status again when a user quit, which avoided server stays in one status forever.
     5. Improved the goal logic which might cause ball stops before it is shown to hit the door (may due to performance issues).
     6. The server now will broadcast the average position of players and ball for clients to synchronize after clients are reporting different position when one turn is end.
  10. 04.27: Programming
      1. The limitation of fps (mentioned in 04.24) can’t eliminate serious issues of asynchronized position. Simply computing the average position can’t solve the problem. In this case, using server to compute all physics movement and then broadcast all location is a relatively simple and more possible way to avoid asynchronization.
      2. Due to various computing performances, the physics calculation of arcade physics, which is a light-weight physic engine, can't be simply repeated by inputting the same attributes and values. As a result, a p2p architecture may not be accepted because of many potential errors caused by physic engine and synchronized issues.
      3. The new server uses Phaser 3 in ‘HEADLESS’ mode, which won’t render objects and is able to run on the server. The ‘datauri’, ‘node-canvas’, and ‘jsdom’ is added to ‘cheat’ Phaser in order to run on server.
      4. The server will now broadcast the position with the timestamp to all clients in every update loop. The client will move the ball and players to the position within the time which is the one between current and last timestamp. Earlier send but later receive position will be give up.
  11. 04.30: SoBT meeting
      1. https://lernraum.th-luebeck.de/mod/forum/discuss.php?d=52551#p66298
  12. 05.04: Debug
      1. The clients now set the position of all objects instead of moving it to the position. The update rate is the same as the fps on the server side, which is 25 frames/loops per second, so simply setting the position is more convenient and don’t cause glitching. It also enables to totally cancel the physic calculation on client side, which again greatly reduces the calculation.
      2. The goal detection will be totally run by server now, it won’t wait for clients respond before send the goal signal. On the other hand, the turn detection and start detection include the confirmation between all clients and server.
      3. The simple goal effect is added.