Weekly Diary

Master thesis course in Computing Science

Pina Kolling

Week 3	Introduction and first work on project plan
Week 4	Finish project plan, start setting up code on my computer
Week 5	First research on the topic, including finding literature, set up Git and LaTeX for master thesis (on work laptop, my laptop and stationary pc), document execution of code
Week 6	Set up code on my computer and first familiarizing with codebase, finding literature, document execution of code
Week 7	Researching options of melt framework (implementing, documenting the process and literature research)
Week 8	Implementing, documenting the process and literature research and vacation with my grandmother (she turns 90 ♡) so probably reduced work capacity
Week 9	Implementing, documenting the process and literature research
Week 10	Implementing, documenting the process and literature research and creating slides for the midterm seminar, evaluating if it is possible to obtain colour-corrected video results using JIT and then specify or readjust the focus
Week 11	Implementing, documenting the process and literature research, midterm seminar
Week 12	Implementing, documenting the process and literature research, search or implement offline colour correction software and other suitable solutions for comparison (if needed)
Week 13	Implementing, documenting the process and literature research
Week 14	Implementing, documenting the process and literature research
Week 15	Writing
Week 16	Writing
Week 17	Writing
Week 18	Writing
Week 19	Finalizing, reworking and applying feedback
Week 20	Hand in final version of the thesis
Week 21	Create Slides for the thesis seminar
Week 22	Thesis seminar (defence and opposition)
Week 23	Opponent thesis report

Week 3

16.01.24, Tuesday

- First meeting at university

17.01.24, Wednesday

- Setting up file and git for weekly diary
- Writing first mail with topic specification to Vicenç Torra
- Keeping my supervisor at Codemill (Urban Söderberg) in the loop
- Begin with project plan (setting up the file, etc.)

18.01.24, Thursday

- Getting a supervisor from university assigned (Cem Okulmus)
- Continue work on project plan:
  - Introduction
- First research on:
  - Just-In-Time (JIT), WebRTC, h.264, Melt framework
  - Infrastructure model of the system

19.01.24, Friday

- Continue work on project plan:
  - Problem formulation
  - Method
  - Infrastructure model

20.01.24, Saturday

- Continue work on project plan:
  - Evaluation methods
  - Self assessment
- Looking into previous master thesis that was written at Codemill

Week 4

Info: The Codemill logo marks the days at which I have been at the company's office.

22.01.24, Monday

- Set up git on other computer
- Continue work on project plan:
  - Resources
  - Read again and correct
  - Deciding on a title
- Send projectplan to supervisor at Codemill (Urban Söderberg)
- Send projectplan to supervisor at university (Cem Okulmus)

23.01.24, Tuesday

- First meeting with supervisor at university (Cem Okulmus)
- Rework and additional info on project plan:
  - Change JIT definition
  - Add timeline
  - Add challenges
- Add timeline weekly diary and adapt setup of weekly diary (counting in calendar weeks)

24.01.24, Wednesday

- Prepare laptop to set up code on it

25.01.24, Thursday CODEMILL

- Setting up the code on my laptop at Codemill

  (generating ssh key, cloning git repositories, installing node.js and docker, etc)
  - Problem: My RAM was not sufficient and the code could not be executed
  - Solution: Looking for a company laptop to execute the code

26.01.24, Friday CODEMILL

- Setting up the code on the new laptop at Codemill
  - Problem: Space in user name on the device which makes some paths not working
  - Solution: Setting up windows with a new user (to do)
  - Info: The code has not been run on a windows system before

Week 5

29.01.24, Monday

- Being sick ☹

30.01.24, Tuesday

- Being sick ☹

31.01.24, Wednesday

- Being sick ☹
- Setting up new windows user
- Setting up code on new laptop (frontend running but problems with backend/docker container)
- Document execution of code:

Setting up the code

- Generate ssh key (`ssh-keygen`) and add to GitLab
- Clone git repositories (jit-webrtc and accurate-player-3-core)
- Install node.js and set path variables for npm (and yarn)
- Install and run docker
- Execute jit-webrtc code with command from README with `docker/main/main.sh --threads 16 --port 8080 $VIDEOFILE` (not working!)
- Execute accurate player code (run `npm install --force`, `npm install yarn` and then npm start, resolve errors, fix dependenciy problems with `npm audit fix --force` (potentially twice))

01.02.24, Thursday

- Being sick ☹
- Installing slack
- Looking into the backend/docker problem
- Setting up WeeklyDiary git and tex file on Codemill-laptop

02.02.24, Friday

- Being sick ☹
- Trying to solve the docker/backend problem (still unsolved)
- Setting up git and tex file for master thesis on stationary PC
- Creating title page
- Structure for thesis
- First research and adding of references
- First writing in introduction

03.02.24, Saturday

- Being sick ☹
- Trying to solve the docker/backend problem (still unsolved):
  - Inspecting `main.sh` script file
  - Inspecting docker problems regarding windows
  - `docker-run.sh` not found or opened... Changing the path does not seem to help and the file does exist (feedback: no such file or directory)
  - Setting up python

Week 6

05.02.24, Monday CODEMILL

- Run backend/docker (finally!):
  - Make changes in `main.sh` (last line): remove `--device /dev/fuse` and change path to `//opt/jit-webrtc/jit/docker-run.sh`
- Problem: Connectivity issues between browser and docker
- Solution: Installing Linux and not running it under Windows

06.02.24, Tuesday

- Installing Linux Ubuntu 22.04 (not booting after updates)
- Installing Linux Ubuntu 23.10 (does not work at all)
- Researching and writing an introduction about Codemill
- Installing Linux Ubuntu 22.04
  - The problem originated from the NVIDIA graphics card. Before updating, the drivers had to be installed with `sudo ubuntu-drivers autoinstall`.
- Installing docker, node.js, git, miktex, texstudio and cloning repositories
- Adding to weekly diary: Codemill logo for each day I was at the company's location
- Executing frontend
- Executing backend in docker container

07.02.24, Wednesday CODEMILL

- Connecting backend and frontend
- Running the code
- Setting docker timeout from 15s to 150s in `main.py`
- Create private git repositories to store work progress
- Research on WebRTC and transcoding and looking into code of JIT-WebRTC
- Adding labels and references to structure of master thesis tex file
- Adding README files of code base to master thesis tex file

Running the code

- Frontend:
  - Open folder `accurate-player-3-core/packages/demo` in terminal
  - Execute `JIT_BACKEND=http://localhost:8080 yarn start` or `./start.sh`
- Backend:
  - Open folder `jit-webrtc` in terminal
  - Execute `docker/main/main.sh --threads 16 --port 8080 https://s3.eu-central-1.amazonaws.com/accurate-player-demo-assets/timecode/sintel-2048-timecode-stereo.mp4`
- Open `http://localhost:5000/controls/jit/index.html` in browser

08.02.24, Thursday

- Looking into the backend code, README and the system's components, summarizing and taking notes in the thesis file:
  - Audio Video Interleave (AVI)
  - Named pipe
  - Create diagram of system
  - Python documentation
  - Web services and REST API
- Structure of the thesis

09.02.24, Friday

- Looking into the backend code and the system's components, summarizing and taking notes in the thesis file:
  - Docker container
- Looking into the frontend code and README, summarizing and taking notes in the thesis file:
  - Node.js, yarn and npm

Week 7

12.02.24, Monday CODEMILL

- Write to do list for the next steps and update time schedule
- Look into MLT FX and integration of OpenGL and GLSL?
- Looking into suitable filters (aka plugins) in melt
  - Maybe suitable: `avfilter.colorbalance`, `avfilter.colorchannelmixer`, `avfilter.colorcontrast`, `avfilter.colorlevels`, `avfilter.colortemperature`, `frei0r.coloradj_RGB`, `frei0r.colorize`
  - Probably not suitable: `avfilter.colorcorrect`, `avfilter.colorhold`, `avfilter.colorize`, `avfilter.colorkey`, `avfilter.colormatrix`, `avfilter.colorspace`, `frei0r.colordistance`, `frei0r.colorhalftone`, `frei0r.colortap`, `frei0r.three_point_balance`, `frei0r.contrast0r`, `tcolor`

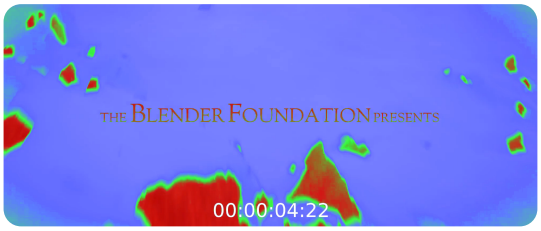
To Do

- Figure out, where the colour grading should be implemented
  1. Does melt already have an option for this?
  2. Can it maybe only be done when the video is paused?
  3. Is there a different place in the system, where the colour grading can be done?

13.02.24, Tuesday CODEMILL

- Looking briefly into `melt.c`, `JitControl.proto`, `JitStatus.proto` and other melt files to find out, where to attach a filter/plugin to a video and where the quality setting is changed
- Getting first overview over structure of melt
- Execute melt with filter without JIT to test the filters: `melt https://s3.eu-central-1.amazonaws.com/accurate-player-demo-assets/timecode/sintel-2048-timecode-stereo.mp4 -filter avfilter.colorbalance av.rs=1 av.gm=1 av.bh=1` for intense colours

Original colours



Colours with `av.rs=1 av.gm=1 av.bh=1`

→ This can be used for the offline comparison

- Looking into `local_melt.py` and `main.py`

14.02.24, Wednesday CODEMILL

- Looking into `local_melt.py` and `main.py` and trying to add command for melt there
- On the trail of print statements - disappeared or user error? (Aka figuring out where the logging info and print statements are printed to get more insight of the code)

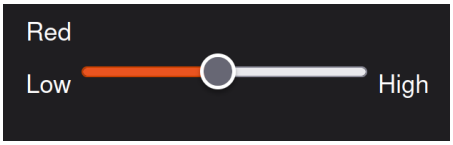
→ Add `-v` to command to see logs in the terminal: `docker/main/main.sh -v --threads 16 --port 8080 https://s3.eu-central-1.amazonaws.com/ accurate-player-demo-assets/timecode/sintel-2048-timecode-stereo.mp4`

- Fun fact (might be useful later): Find the name of your docker container with `docker ps` and get info with `docker logs --follow <container-name>` (but the missing info cannot be found there either so far)
- Changing overall input command for melt and adding a video filter to the video, that can be viewed in the accurate player using JIT 🎉

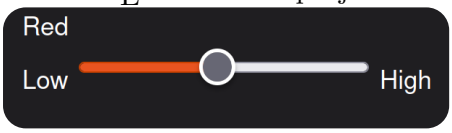
- Add filter to `melt_args` in different place in code
- Start looking into main.py and frontend code base to implement slider to change colour intensity and add different options from the frontend

16.02.24, Friday

- Look into frontend code
- Add slider for colour red in `packages/demo/src/controls/jit/index.html` (without any backend functionality so far):



- Small LaTeX `tikz` side project for rounded corners of the graphics in my documents:



18.02.24, Sunday

- Downloading ressources to prepare offline work during travel
- Trying to get docker running with local video files (problem with `google-crc32c`)

Week 8

19.02.24, Monday

- Reading the accurate player code and following the data flow of the input of the quality slider, especially in
  `packages/demo/src/controls/jit/index.html`,
  `packages/demo/src/controls/jit/JITDemo.ts`,
  `packages/jit/dist/index.d.ts`,
  `packages/jit/src/JITService.ts`,
  `packages/demo/src/controls/jit-session/JITSessionDemo.ts` and
  `packages/core/dist/index.d.ts`

23.02.24, Friday

- Implement feedback in the thesis file:
  - *Related work* before *Structure*
  - Merging Chapter 2 and 3 into a *Preliminaries* Chapter
  - Merge the Subsections in Chapter 2 and 3
  - Remove Subsection *Implementation Details* (Chapter 5)
  - Rename Chapter 5 to *Experimental Evaluation and Discussion*
  - List of Tables and List of Figures
  - References: Use footnotes for the URLs (f.ex. Codemill Website) and remove them from reference list:
    - Define command `\myfootcite`
- Add caption and short caption for list of figures to each graphic
- Define command `\cutpic` for rounded corners in graphics and apply this to the thesis file and the weekly diary

  → Because of very restricted internet, the work on the implementation of the colour correction cannot really be continued (code cannot be executed here) and this is why less important tasks like design of the thesis are done now
- Seperate multiple footnote citations with a comma (using `\textsuperscript{,}` to maintain the correct font of the comma)
- Write on thesis: Motivation and Research Questions

Week 9

28.02.24, Wednesday

- Working on implementation – problems to run the code

→ Suspecting my mother's weird and bad internet setup to be the cause of the problem and hoping, that it will run when I return to Sweden tomorrow.
(Fun fact: Germany has very bad internet.)

29.02.24, Thursday

- Looking into the data path of the quality parameter in the accurate player code for better understanding:
  - → `quality-slider` in `packages/demo/src/controls/jit/index.html` gets input
  - → input value is read out in `packages/demo/src/controls/jit/JITDemo.ts` and then `player?.api.setQuality(value)` is called
  - → `player` has type `JITPlayer` and `api` has type `JitService`
  - → class `JitService` extends class `Service`

01.03.24, Friday CODEMILL

- Making some changes in the schedule and informing supervisors via mail
- Adding red value in different places in the code and test the results for better understanding:
  - Backend: Adding red value with `"av.rs=%.1f" % kwargs['av.rs']` to have a variable that can (hopefully) be changed from the frontend and then the red value will maybe update when it is changes (in `jit-webrtc/jit/local_melt.py`)
  - Adding `RED_VALUE = 1` and `**("av.rs": RED_VALUE)` in `jit-webrtc/jit/main.py` as a parameter to `execute_local_melt` so it is contained in `kwargs['av.rs']` in `jit-webrtc/jit/local_melt.py`
  - Add value for `rs` in `jit-webrtc/jit/ffprobe.py`
  - → This seems to be the red value with which the video gets send initially from melt
- Analysing line ∼180-200 in `jit-webrtc/jit/main.py` to add the red value there for it to be able to be updated:
  - How do I get `encoder = videosender._RTCRtpSender__encoder` to return the red value?
  - Adding *red* in line ∼180-200
  - Adding *red* in line ∼530

Week 10

04.03.24, Monday CODEMILL

- Red value can be changes when starting docker with `--rs <value>`
- `json.dumps()` function will convert a subset of Python objects into a json string
  → add red value to be send to melt in line ∼180-200 in `jit-webrtc/jit/main.py`
  → retrieve red value from frontend in line ∼180-200 in `jit-webrtc/jit/main.py`
- Where does the data get send to melt? Where does melt receive it? Where is it processed?
  - `local_melt.py`? → Seems to be for initialization
  - Reading through `README`s in subfolders:

    Data Flow

    Commands are sent from the browser to `main.py` via a WebRTC data channel, and then from `main.py` to `melt` via a `stdout/stdin`.

    `melt` in turn sends an AVI stream with rendered video and audio to `main.py` via a named pipe, and status messages and metadata is sent to main.py via another named pipe.

  - Looking into `class StdinVideoTrack(VideoStreamTrack):` in line 393 in `main.py`
  - Trying to send input of quality slider to main.py:
    - Adding `getRedSliderValue()` in `packages/demo/src/controls/jit-session/JITSessionDemo.ts`