Project Information

Title: Live Object Pattern Recognition and Color Detection
Team: 2 developers/team preferred, single work on request
Deadline: KW44 *(can be shifted if needed)*

Introduction

In this project, you will gain hands-on experience by developing a small Python application that uses the livestream of a camera and performs real-time object pattern recognition and color detection. The recognition is visualized within the livestream and the information gathered is logged into a csv file for later analysis. The goal is to learn the basics of software development including following aspect:

- **Dependency management**: Gain experience in handling external libraries and packages
- **Architecture and Code Structuring**: Develop a well-structured and modular codebase that uses to best practices in software architecture.
- **Documentation**: Improving your skills in writing clear documentation including static and runtime-view diagrams.
- **Version control systems**: Familiarize yourself with version control systems like git and start working with issues, merge requests co.

Must haves

Features

- **Object Pattern Recognition:** The system should be able to detect patterns like circle, rectangle, square, triangle.
- **Color Detection:** Inside each pattern, the system shall determine the main color of the pattern. Following colors should be detectable: red, green, blue, yellow, violet.
- **Real-time Visualization:** Overlay recognized objects with their respective patterns and colors on the live video/image.
- **Data Logging:** Log timestamp, pattern type, detected color, and any other relevant information into a structured CSV file.

Documentation and others

- **Dependency Management:** Solve the dependency management problem, so everybody can run the code with ease.
- **Version Control System:** Use a version control system like git (in combination with gitlab) to ensure proper version and release controlling. Use issues, merge requests and feature branches with peer-to-peer reviews.
- **Clean Code Principles:** Use of advised clean code principles while writing your code.
- **Architecture:** Organize your codebase into modular components for maintainability and extensibility. Create classes/modules with distinct responsibility. Create a component diagram with all components and their relation to each other. The architecture shall be a part of the repository.

- **Detailed Documentation:** Create documentation to guide users through your applications setup, features, and usage. Please provide a meaningful runtime view diagram showing what your application does. The documentation shall be a part of the repository.

Optional features

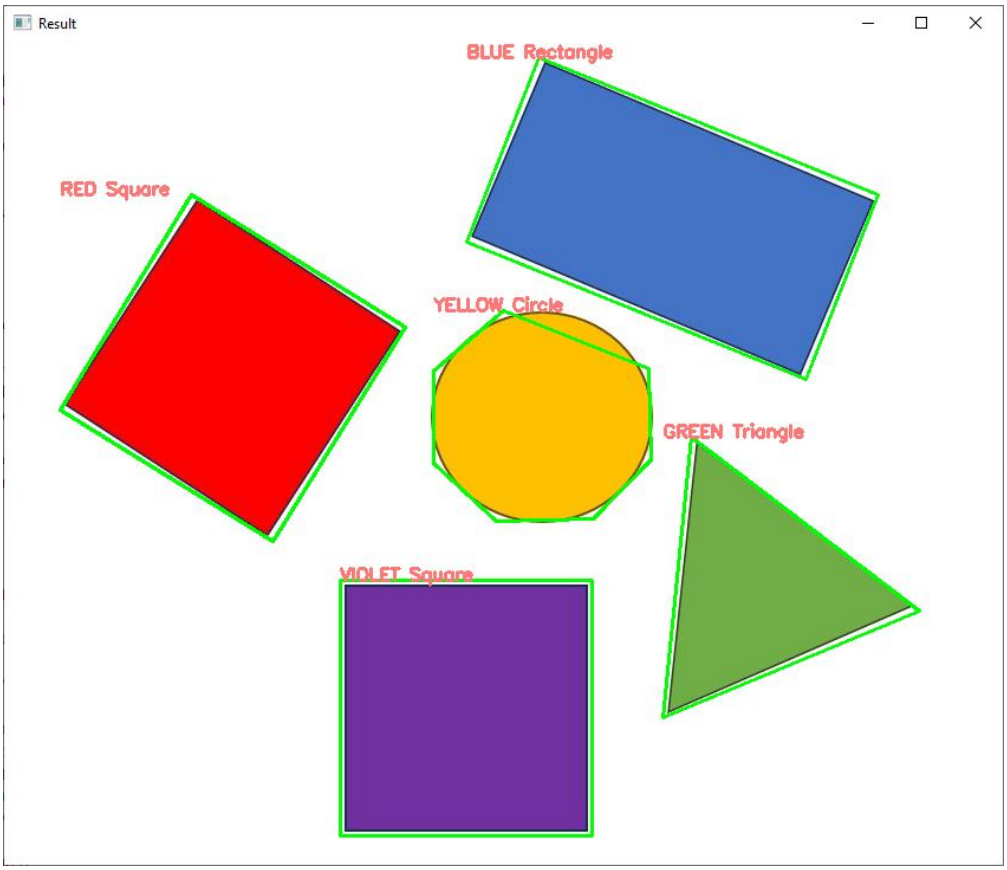
To get a better grade, following features can be implemented:

- **UI:** Selectable ROI using mouse
- **UI:** Log file path selection option
- **FEAT:** Acoustic shape and color output (you can use libraries like gTTS)
- **FEAT:** Application configuration using a .conf file (e.g. Color detection ranges)

Examples

Pattern and color recognition

The provided sample image can be used for testing purposes.



Logging

```
|----------------------------|-----------|--------|
| Timestamp                  | Pattern   | Color  |
|----------------------------|-----------|--------|
| 2023-08-28 17:12:47.090492 | Square    | VIOLET |
|----------------------------|-----------|--------|
| 2023-08-28 17:12:47.093486 | Triangle  | GREEN  |
|----------------------------|-----------|--------|
```

```
| 2023-08-28 17:12:47.094483 | Circle    | YELLOW |
|----------------------------|-----------|--------|
| 2023-08-28 17:12:47.096480 | Square    | RED    |
|----------------------------|-----------|--------|
| 2023-08-28 17:12:47.098484 | Rectangle | BLUE   |
|----------------------------|-----------|--------|
```

Proposed approach

Day 1 (Project setup)

- Create your team, if not already done
- Study the project description and identify requirements. Ask questions if something is unclear.
- Install VSCode, python (if not already done)
- Install "Markdown All in One" and "Markdown Preview Mermaid Support" Extension
- Install poetry and create your project
- Define the folder structure
- Create a gitlab account and your project, invite the team members and me as supervisor
- Make your first push to gitlab repository
- ...

Day 2 (Architecture)

- Start identifying modules and classes
- Document your findings, create a component diagram
- Start structuring your code repository using folders and files.
- Start creating and documenting your concept using mermaid.
- Start coding

Day 3 (Coding)

- Up to you

Day 4

- We have a different program and you will now have time to work on your project

Project Submission

- Link to your GitLab repository
- Project state as .zip file
- Please document implemented features, or additionally implemented features

Questions or support needed?

You can always contact me using this e-mail address: klims.saikins@cedes.com
I will try to give you a respond inner 24 hours.
Small interim review possible on request.