TED UNIVERSITY

SPRING 2024

CMPE 491

High Level Design Report

Project Name

3DifyMe

Project Supervisor

Yücel Çimtay

Project Team Members

Hamza Emin Hacıoğlu

Deniz Caner Akdeniz

Bengisu Tuğrul

Elif Arslan

Project Jury Members

Gökçe Nur Yılmaz

Müslim Bozyiğit

Contents

1 Introduction

1.1 Purpose of the system

The primary purpose of this project, titled '3DifyMe,' is to develop a sophisticated/well-designed desktop application that transforms conventional 2D videos and live camera streams into immersive 3D experiences. We achieve this primarily through the strategic use of color theory and optical illusions. By manipulating colors, saturation, and contrasts, we can create the illusion of depth and dimension within the 2D image, fooling the human brain into perceiving a 3D world. As mentioned in the article, “Unsupervised generation of high-quality multi-view consistent images and 3D shapes using only collections of single-view 2D photographs has been a long-standing challenge.” [3]. To overcome this challenge, Generative Adversarial Networks (GANs) will be employed as a secondary tool or helping hand to further refine and enhance the depth created through color theory. This combined approach will significantly increase the user's visual experience with heightened depth perception and a greater sense of realism.

'3DifyMe' bridges the gap between traditional 2D media and cutting-edge 3D technology, providing users with a seamless transition into three-dimensional content. This system is developed with a strong commitment to coding ethics and best practices, aiming to advance the field of image processing while ensuring a smoother and responsible user experience.

The purpose of this system can be summarized as follows:

- To develop a user-friendly application capable of converting 2D videos and live streams or 2d media into 3D in real time.
- To utilize deep learning techniques to improve the accuracy and realism of the 3D conversion.
- To provide an intuitive and accessible interface that allows users to interact with the application effortlessly.
- To enhance the overall user experience by adding visually appealing depth effects to media content.

1.2 Design goals

The design objectives of the "3DifyMe" project revolve around creating a resilient, efficient, and user-centric (easy to use) application. These objectives ensure that the system meets both functional and non-functional requirements while delivering an exceptional user experience. The design goals include:

1. **Real-Time Conversion**:
    - Enable real-time conversion of 2D videos and live camera streams into 3D. This requires optimizing processing algorithms to minimize latency and ensure smooth performance. This is efficient because optimized algorithms minimize latency, ensuring smooth performance.
2. **User-Friendly Interface**:
    - Develop an intuitive and easy-to-navigate graphical user interface (GUI). It should include clear instructions, tooltips, and an organized layout to facilitate user

interaction. It provides an intuitive GUI enables swift navigation and interaction for better efficiency.

3. **Enhanced User Experience**:
    - Focus on improving the visual appeal and realism of converted 3D content. This involves dynamically adding depth layers and providing options for users to adjust depth levels according to their preferences. It improved visual appeal and realism encourage user engagement for better efficiency.
4. **Customization**:
    - Allow users to customize various settings within the application, such as video resolution, frame rate, and processing quality. Customization options should cater to different user needs and device capabilities for better efficiency.
5. **Comprehensive Features**:
    - Integrate additional functionalities to enhance the application's utility. These features may include camera controls (e.g., on/off, live processing), mirror view toggling, video recording, downloading capabilities, and a detailed "How to Use" section for user guidance. Additional functionalities enhance versatility and usability for better efficiency.
6. **Performance Optimization**:
    - Ensure efficient operation across various hardware configurations. Implement adaptive processing quality settings to automatically adjust parameters based on the user's device capabilities, balancing performance and visual quality. For better efficiency, adaptive quality settings optimize performance across devices.
7. **Offline Rendering**:
    - Provide offline rendering capabilities for users without stable internet connections. This feature allows 2D videos to be converted to 3D without requiring an internet connection, ensuring accessibility and convenience. For better efficiency, it provides accessibility and optimizes processing times for users with limited internet access.
8. **Performance:**
    - Optimize the application for fast and efficient processing across various hardware configurations, including lower-end devices. It ensures fast and efficient processing across various hardware configurations.
9. **Flexibility and Control:**
    - Implement adaptive processing to automatically adjust settings based on device limitations.
    - Allow for offline video conversion and exporting for users with limited internet access. Adaptive processing adjusts settings based on device limitations.
10. **Future-Proofing:**

- Design the application with scalability in mind to handle increasing user loads and data volumes.
- Ensure the application can integrate with potential future technologies like Augmented Reality (AR). Scalable design handles increasing user loads and data volumes.

11. **Security and Reliability:**

- Prioritize secure data handling practices like encryption and authentication to protect user information.
- Implement rigorous testing procedures to identify and address potential bugs and errors, ensuring application reliability.
- Provide comprehensive user documentation and code documentation (for developers) for troubleshooting and future enhancements. Secure data handling and rigorous testing ensure reliability and user data protection.

1.3 Definitions, acronyms, and abbreviations

To ensure clarity and consistency throughout the document, the following definitions, acronyms, and abbreviations are used:

- **2D (Two-Dimensional)**: Refers to media that has height and width dimensions only.
- **3D (Three-Dimensional)**: Refers to media that includes height, width, and depth dimensions, creating a perception of three-dimensional space.
- **Color Theory**: Color theory is the study of how colors relate to each other and how they affect human perception.
- **GAN (Generative Adversarial Network)**: A type of deep learning algorithm consisting of two neural networks, the generator, and the discriminator, that are trained together to produce realistic synthetic data. [6]
- **GUI (Graphical User Interface)**: A user interface that allows users to interact with electronic devices through graphical elements like icons, buttons, and windows.
- **Real-Time**: The capability of processing data and providing output almost instantaneously as the data is received.
- **Depth Layers**: Additional layers added to an image to create the illusion of depth, enhancing the three-dimensional appearance.
- **Deep Learning**: A subset of machine learning involving neural networks with multiple layers that can learn and make intelligent decisions from vast amounts of data.
- **Scalability**: The ability of the application to handle increasing demands without sacrificing performance.
- **User Interface (UI)**: The graphical elements that allow users to interact with the application.

- **User-Friendly**: Easy to understand and navigate, promoting a positive user experience.
- **API**: Application Programming Interface
- **Persistent Data Management:** The process of storing and managing data for extended periods. (Section 3.4)
- **Data Access Layer:** A software layer that simplifies access and manipulation of data stored in a database or file system. (Section 3.4)
- **Event-Driven Architecture:** A software design pattern where events trigger actions within the system. (Section 3.6)
- **Global Software Control:** A central mechanism that manages the flow of data and coordinates actions between different parts of a software system. (Section 3.6)
- **Boundary Conditions:** The limits or edge cases that a system needs to consider to function correctly. (Section 3.7)
- **Image Preprocessing**: Techniques applied to images before processing to improve their quality or prepare them for further analysis. (Section 3.7)
- **Optical Flow:** A technique for estimating the motion of objects in a video sequence. (Section 3.7)
- **Object Segmentation:** The process of dividing an image into regions corresponding to individual objects. (Section 3.7)

1.4 Overview

The "3DifyMe" project aims to revolutionize the way users experience video content by transforming traditional 2D videos and live camera streams into immersive 3D experiences. This section provides a comprehensive overview of the project's objectives, methodology, and features.

Project Objectives: The primary objective of this project is to develop a desktop application that seamlessly converts 2D media into 3D. By leveraging deep learning techniques (there are several core techniques in traditional computer vision as mentioned in the website [1]), particularly GANs, the application aims to enhance depth perception and realism in the converted content. The project also seeks to provide a user-friendly interface that makes this advanced technology accessible to a broad audience.

Methodology: The conversion process involves several key steps:

1. **Video Input**: Users can input either pre-recorded 2D videos or live camera streams into the application.
2. **Image Processing**: The application uses image processing techniques, color theory, and GANs to analyze and process each frame of the video, adding depth layers to create a 3D effect.
3. **Real-Time Conversion**: For live camera streams, the application processes the input in real-time, providing immediate 3D conversion and display.
4. **User Interface**: The GUI allows users to interact with the application, view side-by-side comparisons of 2D and 3D content, and adjust settings as needed.

Features: The application includes a range of features designed to enhance usability and functionality:

- **Camera Controls**: Users can control the camera, start, and stop live processing, and toggle the mirror view.
- **Recording and Downloading**: Users can record 3D videos and download them for later viewing or sharing.
- **Customizable Settings**: Options to adjust video resolution, frame rate, and processing quality ensure the application meets diverse user needs.
- **User Instruction**: Comprehensive instructions and tooltips guide users through the application, with additional help documentation and tutorials available.

Innovative Extensions: The project is committed to continuous improvement and innovation. Potential future enhancements include additional features such as camera control, mirror view toggling, 2D to 3D image conversion, and advanced user guidance.

In summary, the "3DifyMe" project seeks to enhance user experience by providing a seamless and intuitive way to convert 2D media into immersive 3D content. Through advanced image processing techniques and an easy to use design, the application aims to set a new standard in video viewing technology.

2. Current software architecture (if any)

The current software architecture of the application, which converts 2D videos or live camera streams into 3D experiences, is structured as follows:

2.1 Overview

The existing application is desktop-based software designed using Python and the Qt framework. It leverages the OpenCV library for video processing and applies simple 3D filtering techniques to convert 2D media into a 3D format. As mentioned in the website named “Computer Vision Guide” , “OpenCV is a highly optimized library with focus on real-time computer vision applications. The C++, Python, and Java interfaces support Linux, MacOS, Windows, iOS, and Android.” [5]. In our project, Computer Vision Guide is very effective in determining our path and enlightening us. The architecture includes a graphical user interface (GUI) that provides functionalities for users to load videos, start live camera feeds, apply 3D filters, and view the results side by side with the original 2D content.

The explanation of the algorithms that we used or we are very close to implement in our application and their pseudocode respectively :

1- Live Video Processing:

Function to apply red-cyan 3D filter to a video frame:

This function is designed to process individual frames of live video streams.

It takes a single video frame as input.

Firstly, it separates the frame into left and right halves.

Then, it applies a red filter to the left half and a cyan filter to the right half.

The filtered halves are then combined to create a new frame with enhanced 3D effect.

This function is suitable for real-time applications where frames are processed as they are captured from a live video source.

2-Video Input Processing:

Function to check video file type:

This function is primarily used in the context of processing pre-recorded videos stored as files.

It takes the file path of a video as input.

The function extracts the file extension from the provided file path.

Then, it checks whether the extracted file extension corresponds to a supported video format.

If the format is supported, the function returns a success message; otherwise, it returns an error indicating an unsupported video format.

Function to process an input video file:

This function is intended for processing entire video files stored on disk.

It takes the input file path and output file path as arguments.

The function begins by checking if the input video file type is appropriate by calling the checkVideoFileType function.

If the file type is unsupported, it returns an error.

The function then proceeds to open the input video file and retrieves its properties such as frame dimensions and frame rate.

A VideoWriter object is created to prepare for writing processed frames to an output video file.

The function iterates through each frame of the input video, applying the apply3DFilter function to enhance the 3D effect of each frame.

The filtered frames are written to the output video file using the VideoWriter object.

Once all frames have been processed, the input video file and the VideoWriter object are released to free up system resources.

3-Image Processing PsudoCode

Note: There is more than one way to enhance the effect of an image to make it more look like 3d, in previous reports, we frequently talked about GAN(Generative Adversial Network), this psudocode that you see in the below is another approach rely on Depth Map, in the second course of this project we will be able to explore and implement more of these approaches. “MiDaS” model that we used in the below are a pre-trained model to get the depth map of the input image.

1. Load the MiDaS model:
    1. Load the pre-trained MiDaS model using a deep learning framework.
    2. Ensure the model is loaded onto the CPU for inference.

2. Load image transformation functions:
    1. Load necessary image transformations for preprocessing.
    2. These may include resizing, normalization, or other preprocessing steps.

3. Define a function to enhance the 3D effect:
    1. Input: Original image and its corresponding depth map.
    2. Convert the depth map to grayscale.
    3. Apply a color map for visualization.
    4. Blend the original image with the colored depth map.

4. Specify the directory containing images:
    1. Define the path to the directory containing the images.
    2. Ensure the directory contains only image files (e.g., JPEG, PNG).

5. Loop through images in the directory:
    1. Iterate over each image file.
    2. Read the image file.
    3. Convert the image to the RGB color space if necessary.
    4. Apply any required image transformations.
    5. Perform depth prediction using the MiDaS model.
    6. Enhance the 3D effect of the image using the defined function.
    7. Display the original and enhanced images.

6. End of loop.

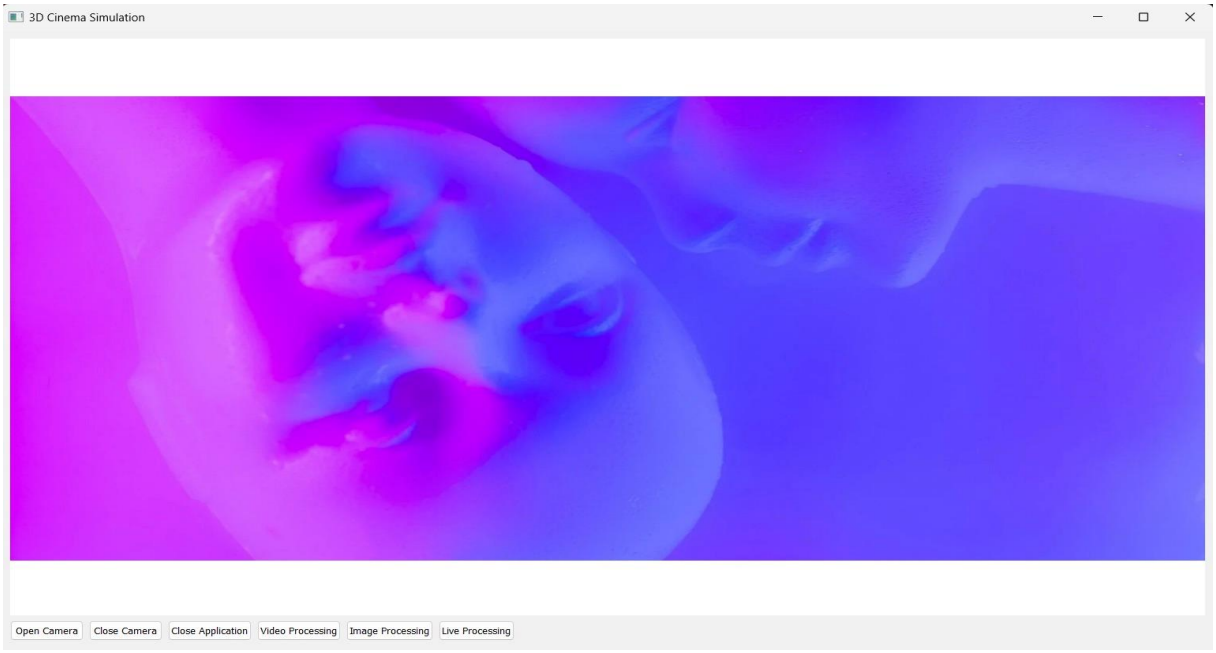
--------- End of Image Processing Algorithm of Ours

Pseudocode of the Live Processing:

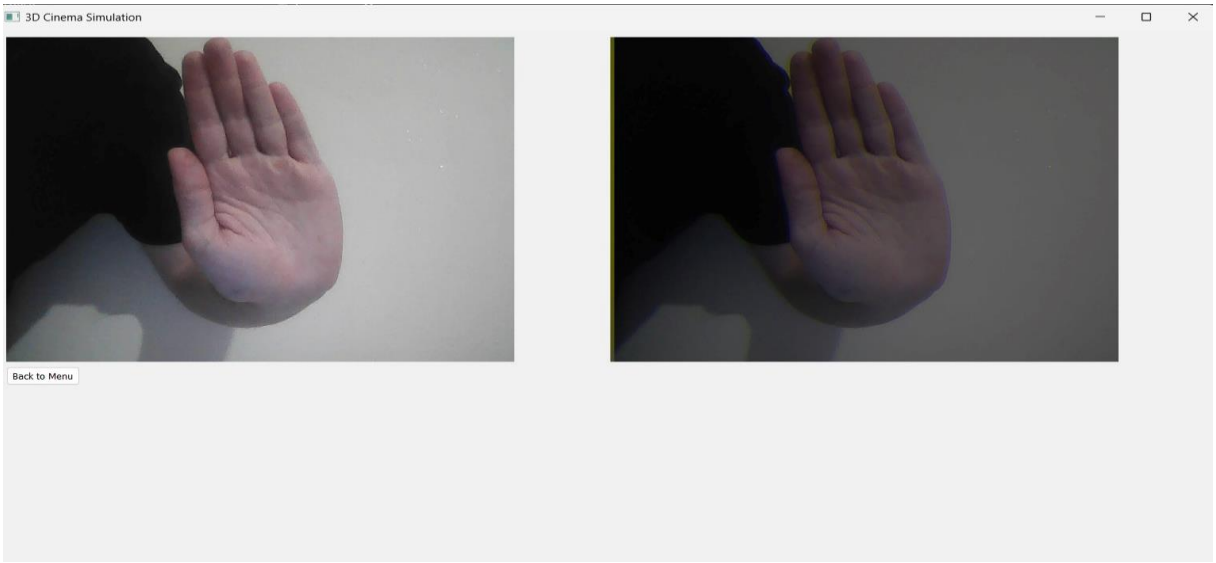
Note: Since the algorithm are too long and not to fill the report with , I am only sharing the shortest one.

```
// Function to apply red-cyan 3D filter to a video frame
function apply3DFilter(frame):
    // Get the dimensions of the frame
    width = frame.width
    height = frame.height
    // Define the region of interest (ROI) for left and right halves
    left_half = frame[:, :width//2]
    right_half = frame[:, width//2:]
    // Create empty images for left and right filtered halves
    left_filtered = createEmptyImage(height, width//2, 3)
    right_filtered = createEmptyImage(height, width//2, 3)
    // Apply red filter to the left half
    for y from 0 to height-1:
        for x from 0 to width//2-1:
            // Extract the pixel values (B, G, R)
            B, G, R = left_half[y, x]
            // Set the red filter: keep the red channel, set green and blue to 0
            left_filtered[y, x] = (0, 0, R)
    // Apply cyan filter to the right half
    for y from 0 to height-1:
        for x from 0 to width//2-1:
            // Extract the pixel values (B, G, R)
            B, G, R = right_half[y, x]
            // Set the cyan filter: keep the green and blue channels, set red to 0
            right_filtered[y, x] = (B, G, 0)
    // Concatenate the left and right filtered halves
    filtered_frame = concatenateImages(left_filtered, right_filtered)
    return filtered_frame
```

Sample screenshots from our application:



Explanation: The menu is in the below, the open camera, close camera and the Live processing buttons are functional. We do realize that the button layout is not optimal, but we will make necessary changes before the presentation of our project.



Explanation: Left video output is non-filtered. And we applied red-cyan filter in the right video output. We just put this screenshot to show, how our project looks like. Also, we do realize that the button layout is not optimal, but we will make necessary changes before the presentation of our project.

2.2 Subsystem Decomposition

The application is divided into several subsystems:

User Interface Subsystem: Built using Qt, this subsystem handles all user interactions and displays. It includes components such as buttons for starting/stopping the camera, loading videos, and switching between the menu and video display pages.

Video Processing Subsystem: Utilizes OpenCV for capturing video frames, applying 3D filters, and rendering the output. This subsystem processes both live camera feeds and pre-recorded videos.

Control Subsystem: Manages the flow of data between the UI and the video processing subsystems. It includes logic for handling user inputs, starting/stopping video processing, and updating the display.

2.3 Hardware/Software Mapping

The software runs on standard desktop hardware with the following components:

CPU: Handles general application logic and the GUI.

Camera: Used for capturing live video feeds.

Storage: For saving and loading video files.

The software architecture relies on:

Python: The main programming language for application logic and video processing.

Qt: For building the GUI and managing user interactions.

OpenCV: For video capture and processing functionalities.

3. Proposed software architecture

3.1 Overview

The proposed architecture aims to enhance the current system by introducing more advanced 3D processing techniques, improving user experience, and ensuring better performance and scalability. The new system will still be a desktop application but with a more modular and efficient design.

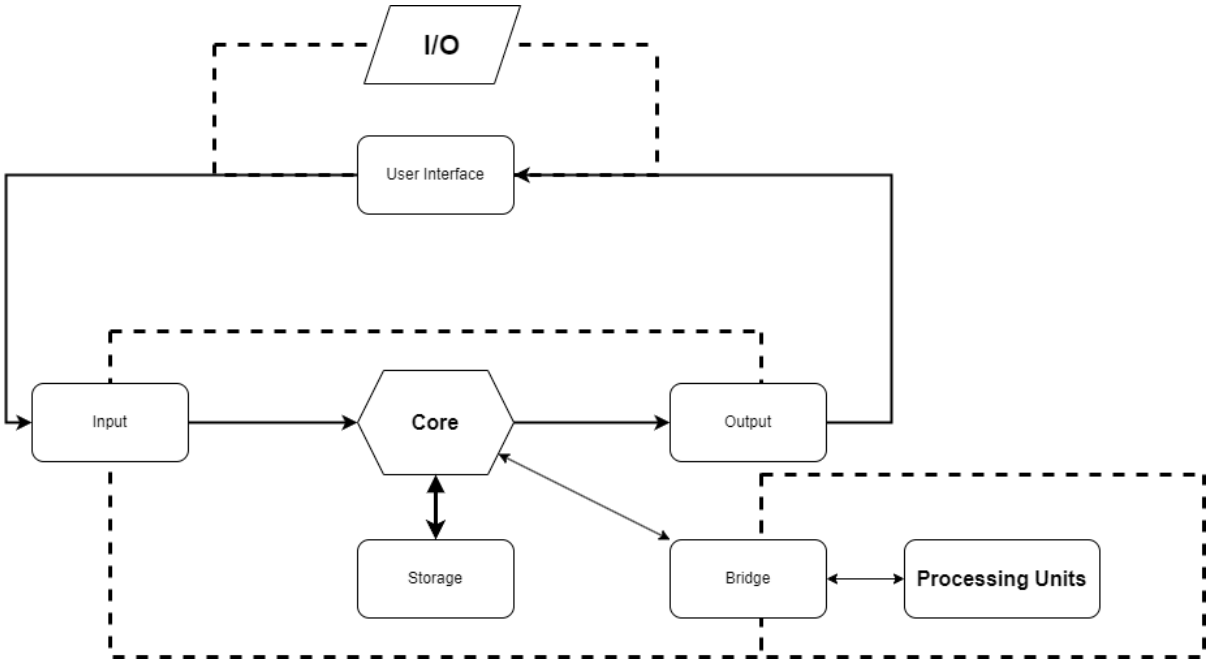
3.2 Subsystem decomposition

Enhanced User Interface Subsystem: Using Qt for a more intuitive and responsive GUI, incorporating additional features such as real-time depth adjustment controls, recording, and playback capabilities.

Advanced Video Processing Subsystem: Utilizing advanced algorithms and possibly machine learning models to create more realistic 3D effects. This subsystem will also support higher resolutions and better performance.

Data Management Subsystem: Handling video file storage, retrieval, and management, ensuring efficient access and modification of media files.

Control and Integration Subsystem: Orchestrating the interaction between UI, video processing, and data management subsystems. Ensuring smooth and efficient data flow and processing.



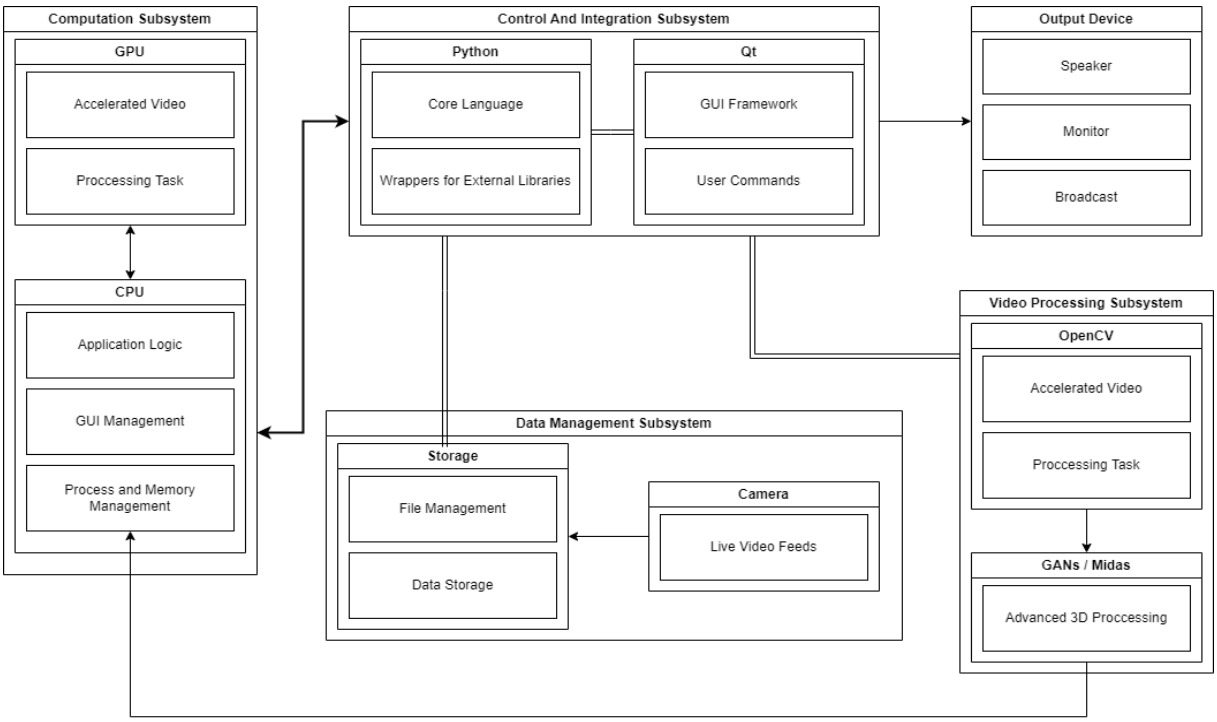
3.3 Hardware/software mapping

The proposed system will leverage both CPU and GPU resources more effectively to ensure smooth real-time processing. It will include:

Multi-core CPUs: For managing complex application logic and user interactions.

GPUs: For accelerating video processing tasks, especially the application of 3D effects.

High-Resolution Cameras: To capture detailed live feeds for better processing.



3.4 Persistent data management

Our project does not generate or process user data during the process of converting 2D videos and live camera streams into 3D experiences. Therefore, we currently do not have a persistent data management need.

However, some intermediate data may be generated during processing. This intermediate data will be automatically deleted after the processing is complete. No user data or intermediate data will be stored permanently to protect user privacy and ensure data security.

Even though the data is not stored, there are some details that need to be taken into consideration during the processing phase:

“The data management process includes a wide range of tasks and procedures, such as:

1. Collecting, processing, and validating data

2.Integrating different types of data from disparate sources, including

3. Protecting and securing data and ensuring data privacy

4. Managing the lifecycle of data, from creation to deletion”

and these important details were created for our own project based on the website.[7]

Data Processing and Inference

Our project temporarily processes user-provided videos and camera images to convert 2D videos and live camera streams into 3D experiences. During processing, intermediate data is generated using image processing and deep learning techniques. This intermediate data is automatically deleted after the processing is complete and is not stored in any way.

User Privacy and Data Security

Our project prioritizes protecting user privacy and data security. Therefore, no user data or intermediate data is stored permanently. All data is automatically deleted after processing is complete. Our system is equipped with appropriate security measures to prevent unauthorized access and data breaches.

Conclusion

Our project is not designed to process or store user data. Therefore, we currently do not have a persistent data management need. To protect user privacy and ensure data security, all data is automatically deleted after processing is complete.

Future Considerations: User Account System

We are considering implementing a user account system with username and password login in the future. If implemented, this will introduce a need for persistent data management to store user data and system settings securely. Here's how we envision handling data management in that scenario:

Data Storage: Persistent data, such as processed 2D and 3D videos, model parameters, user settings, and user credentials, will be stored in a secure storage system like a database or file system.

A database is preferable for organized and searchable storage of user data and settings.

A file system is better suited for large files like videos.

Data Access Layer: A data access layer will be developed to simplify data access and manipulation. This layer will abstract the low-level access to the database or file system, providing a consistent and user-friendly interface for the application layer.

Data Backup and Recovery: Regular data backup and recovery procedures will be implemented to prevent data loss in case of hardware failures or data corruption.

Security: Strong security measures will be taken to prevent unauthorized access and data breaches. This may include data encryption, user authentication, and access control mechanisms.

Benefits of the Anticipated Data Management Approach

Data Security and Integrity: Secure storage and management will minimize the risk of data loss and corruption.

Ease of Data Access: The data access layer will simplify data retrieval and manipulation for application development and maintenance.

Scalability: The data management infrastructure can be easily scaled to accommodate future growth in user base and data volume.

Conclusion (Future State with User Accounts)

Implementing a user account system will significantly increase our data management requirements. Carefully addressing the considerations outlined above will be crucial for safeguarding user data privacy, ensuring data security, and maintaining overall system security.

3.5 Access control and security

Our application prioritizes user privacy and data security. As user accounts and login functionalities are not currently implemented, we rely on strong access control mechanisms to safeguard the integrity of the application and prevent unauthorized access. (e.g., IP address based access control, reference check etc.) This involves measures like restricting application functionality based on user roles (e.g., developer vs. tester) and employing secure coding practices to minimize vulnerabilities. Additionally, the application will be deployed on a trusted platform and regularly updated with the latest security patches to address potential security risks. We are committed to maintaining a secure environment for all users and will continuously evaluate and enhance our security measures as the project evolves.

Role-Based Access Control:

Users are granted access to application functionalities based on their roles and permissions. This restricts access to sensitive information and ensures that only authorized users can perform specific tasks. For instance, a developer can make code changes that a tester cannot, while a tester can perform debugging and testing procedures that a developer cannot.

Regular Security Updates:

We will regularly update our application with the latest security patches and updates. This helps protect against new vulnerabilities and maintain the overall security of the application.

Security Awareness Training:

We will provide security awareness training to our users to educate them about cybersecurity threats and how to protect themselves. This helps users use the application responsibly and report suspicious activities.

Data Access Restrictions:

Access to user data is restricted to only authorized users who need to access it. This helps prevent data leaks and unauthorized access.

Data Storage and Destruction:

User data is stored only for as long as necessary. Data that is no longer required is securely destroyed.

Data Breach Notification:

In the event of a data breach, we will promptly notify users and take the necessary remediation actions.

Additional Considerations:

- User sessions are automatically timed out after extended inactivity.
- Access may be restricted, or accounts suspended if suspicious activity is detected.
- Users can report suspicious activities within the application.

Conclusion:

Application security is our top priority. By implementing the measures listed above, we are committed to protecting user data and ensuring the overall security of the application. We will regularly review and update our security framework and proactively work to mitigate potential risks.

Note: Authentication, login, and data storage are not applicable in this step, hence not referenced.

3.6 Global software control

Our project, which aims to revolutionize the viewing experience of 2D videos and live camera streams by transforming them into immersive 3D experiences, necessitates the management of a complex data flow and processing pipeline. In this context, a global software control mechanism will play a crucial role. The system will adopt an event-driven approach, where user interface interactions and background image processing steps are defined as events.

Event Processing

Events, such as button clicks in the user interface or the selection of 2D videos or camera streams, will send signals to the global control mechanism.

These signals will trigger the event processing module, which will then send instructions to the relevant subsystems.

For instance, the selected 2D video file will be sent to the image processing subsystem, where deep learning algorithms (GANs) will be employed to add depth layers.

Data Flow Management

The processed 3D video data will be streamed to the user interface alongside the original 2D video.

The global control mechanism will manage and synchronize the data flow to provide the user with a seamless experience.

The playback controls in the user interface will also be controlled by the global control mechanism, ensuring synchronized playback of 2D and 3D videos.

Error Handling

In case of errors during the image processing stage, the global control mechanism will notify the user and potentially suggest alternative solutions.

For example, if the selected file is in an incompatible format, the user may be prompted to select a different file or provided with file conversion options.

This event-driven, centralized approach will ensure effective coordination among the system's various components, delivering a smooth and responsive user experience.

3.7 Boundary conditions

In the context of our project, which aims to transform 2D videos and live camera streams into immersive 3D experiences, the process of converting such data requires careful consideration of various boundary conditions. These boundary conditions are crucial for ensuring the system's reliability, and prevention of undesired outcomes.

Common Boundary Conditions and Solutions

Low-Quality Images: Low-quality images, resulting from factors such as low lighting, blurriness, or low resolution, can hinder depth perception and 3D reconstruction processes. As mentioned in the article , “Inferring depth (or disparity) from a single image, however, is a highly under-constrained problem. In addition to depth ambiguities, some pixels in the novel view correspond to geometry that's not visible in the available view, which causes missing data that must be hallucinated with an in-painting algorithm.”[2]. While a data-driven method is used in the article, image preprocessing techniques can be used to address this issue in the project. Operations like noise reduction, sharpening, and contrast enhancement can improve image quality, leading to better results.

Complex Scenes: Scenes with numerous objects and intricate backgrounds pose challenges for 3D reconstruction. In such cases, advanced techniques like optical flow and object segmentation can be utilized. These techniques can aid in creating more accurate 3D models by distinguishing objects from the background.

Moving Objects: Moving objects can introduce inconsistencies in 3D reconstruction. To tackle this, methods involving tracking moving objects and excluding them from processing by removing them from the 3D model can be employed.

Real-Time Processing: Generating 3D experiences from live camera streams necessitates real-time processing capabilities. This can be a computationally demanding task, requiring high-performance hardware and optimized algorithms.

Conclusion

During the development of our project, comprehensive testing using various test scenarios, including those encompassing boundary conditions, is essential. These tests will help evaluate the system's ability to deliver the desired performance under diverse conditions.

Addressing boundary conditions will ensure that our project can reliably transform 2D videos and live camera streams into 3D experiences across a wide range of imaging conditions and environments.

4. Subsystem services

In our project, we have decomposed the system into several subsystems to ensure modularity and clear separation of concerns. Each subsystem provides specific services that contribute to the overall functionality of the '3DifyMe' application. Below is an overview of the subsystem services:

4.1 User Interface Subsystem

Services Provided:

Video Input Selection: Allows users to select the source of the video input, either from a file or a live camera feed.

Filter Selection: Provides options for users to choose different 3D filters and preview them in real time.

Side-by-Side Display: Enables the display of the original 2D video alongside the 3D converted video, allowing users to compare the results.

Control Panel: Includes playback controls, depth adjustment sliders, and buttons to start or stop the 3D conversion process.

Error Notifications: Displays error messages and suggestions for corrective actions if any issues occur during video processing.

4.2 Video Processing Subsystem

Services Provided:

2D to 3D Conversion: Uses advanced image processing algorithms and deep learning techniques (especially GAN-based image generation as the example named “3DGEN” in the article [4]) to convert 2D videos into 3D.

Depth Layer Creation: Analyzes the 2D video frames and generates depth maps to simulate the 3D effect.

Real-Time Processing: Ensures that the 3D conversion happens in real time, especially for live camera feeds.

Quality Enhancement: Applies filters to enhance video quality, such as noise reduction, sharpening, and contrast adjustment, to improve the 3D effect.

4.3 Data Management Subsystem

Services Provided:

File Management: Handles the loading and saving of video files, including support for various video formats.

Temporary Data Storage: Manages intermediate data generated during video processing, ensuring it is efficiently stored and deleted after use to maintain performance.

Settings Management: Saves user preferences and application settings to ensure a consistent user experience across sessions.

4.4 Control and Integration Subsystem

Services Provided:

Event Handling: Manages events triggered by user interactions, such as button clicks and video selection, and directs these events to the appropriate subsystems.

Subsystem Coordination: Ensures smooth communication and data flow between the user interface, video processing, and data management subsystems.

Error Handling and Recovery: Detects and handles errors that occur during the processing, providing feedback to the user and attempting recovery actions when possible.

4.5 Hardware/Software Interaction Subsystem

Services Provided:

Hardware Utilization: Leverages CPU and GPU resources for efficient processing, particularly utilizing GPUs for intensive video processing tasks.

Peripheral Integration: Supports integration with external cameras for live video input and ensures compatibility with various hardware setups.

Performance Monitoring: Continuously monitors system performance to optimize the use of hardware resources and maintain real-time processing capabilities.

By dividing the system into these subsystems and clearly defining the services each one provides, we ensure that our application is modular, scalable, and maintainable. Each subsystem can be developed and tested independently, facilitating a smoother development process and enabling easier troubleshooting and future enhancements.

5. Glossary

- **2D Video**: A video format that displays only height and width dimensions, lacking depth.
- **3D Video**: A video format that adds a depth dimension to height and width, creating a perception of three-dimensional space.

- **Depth Layers**: Layers added to an image to simulate depth, enhancing the three-dimensional effect.
- **Deep Learning**: A subset of machine learning that uses neural networks with many layers to learn from large amounts of data.
- **Generative Adversarial Network (GAN)**: A machine learning model where two neural networks compete against each other to generate realistic data. [6]
- **Graphical User Interface (GUI)**: A visual interface that allows users to interact with software through graphical icons and visual indicators.
- **Real-Time Processing**: The capability to process and output data almost instantaneously as it is received.
- **Artificial Intelligence**: Artificial Intelligence (AI) is the ability of a computer or machine to mimic human intelligence
- **Machine Learning**: Machine learning (ML) is a type of artificial intelligence (AI) that allows computers to learn from data without being explicitly programmed.
- **Multi-View Display**: Presenting two or more different views of the same content simultaneously.
- **Offline Rendering**: Processing video data without an internet connection.
- **Performance**: The speed and efficiency with which the application executes tasks.
- **Pseudo-Requirements**: Considerations that influence the design and development process but are not strictly technical requirements.
- **Scalability**: The ability of the application to handle increasing demands without sacrificing performance.
- **User Interface (UI)**: The graphical elements that allow users to interact with the application.
- **Color Theory:** Color theory is a vast field encompassing the science and art of using color. It delves into: Color Mixing: Understanding how primary, secondary, and tertiary colors interact to create new hues. (Primary colors: red, yellow, blue) (Secondary colors: green, orange, purple)
    - **Color Harmony**: Knowing how to combine colors aesthetically pleasing ways, using concepts like complementary colors, analogous colors, and triadic colors.
    - **Color Psychology**: Exploring how colors evoke emotions and influence human perception. For example, red is often associated with excitement or danger, while green can represent calmness or growth.
    - **Optical Illusions**: Employing strategic color placement and manipulation to create the illusion of depth and dimension within a 2D image.
- **User-Friendly**: Easy to understand and navigate, promoting a positive user experience.
- **Persistent Data Management**: The process of storing and managing data for extended periods. (Section 3.4)

- **Data Access Layer:** A software layer that simplifies access and manipulation of data stored in a database or file system. (Section 3.4)
- **Event-Driven Architecture**: A software design pattern where events trigger actions within the system. (Section 3.6)
- **Global Software Control:** A central mechanism that manages the flow of data and coordinates actions between different parts of a software system. (Section 3.6)
- **Boundary Conditions**: The limits or edge cases that a system needs to consider to function correctly. (Section 3.7)
- **Image Preprocessing**: Techniques applied to images before processing to improve their quality or prepare them for further analysis. (Section 3.7)
- **Optical Flow**: A technique for estimating the motion of objects in a video sequence. (Section 3.7)
- **Object Segmentation**: The process of dividing an image into regions corresponding to individual objects. (Section 3.7)
- **Bitrate**: The amount of data used per second to represent a video, impacting file size and video quality. Higher bitrates generally mean better quality but larger files.
- **Frame Rate (fps):** The number of still images (frames) displayed per second, influencing video smoothness. Common frame rates include 24fps for movies, 30fps for TV shows, and 60fps for high-quality videos.
- **Resolution**: The number of pixels displayed horizontally and vertically, determining video detail. Common resolutions include 720p (1280x720), 1080p (1920x1080), and 4K (3840x2160).
- **Aspect Ratio**: The ratio of the width of the video to its height, such as the common 16:9 widescreen format or the older 4:3 format used in standard-definition televisions.
- **Container Formats (media container formats):** function as standardized digital wrappers encapsulating the compressed audio and video data streams alongside the codecs and metadata necessary for playback.
    - **MP4 (MPEG-4 Part 14)**: A widely used container format that holds video and audio data, often used for online videos and compressed files.
    - **MOV (Apple QuickTime Movie):** Developed by Apple, MOV is another popular container format that can store video, audio, subtitles, and other data.
    - **AVI (Audio Video Interleave**): An older container format commonly used on Windows systems for storing video and audio data.
    - **MKV (Matroska):** A flexible container format that can hold various video and audio codecs, subtitles, and chapters.
- **Codec (Coder/Decoder**): Software that compresses video data for storage and transmission, then decompresses it for playback.
    - **H.264 (AVC/MPEG-4 AVC)**: A highly efficient video codec widely used for online streaming, Blu-ray discs, and many other applications.

- **H.265 (HEVC)**: The successor to H.264, offering even better compression for even higher quality video at lower bitrates.
- **VP9**: An open-source video codec developed by Google, often used for web video streaming.
- **AV1**: Another open-source video codec gaining traction, offering high compression efficiency like H.265.
-

6. References

[1] Alvi, F. (2024) Deep Learning for Computer Vision: Essential Models and practical real-world applications, OpenCV. Available at: https://opencv.org/blog/deep-learning-with-computer-vision/ (Accessed: 27 May 2024).

[2] Xie, J., Girshick, R., & Farhadi, A. (2016). Deep3D: Fully Automatic 2D-to-3D Video Conversion with Deep Convolutional Neural Networks. arXiv preprint arXiv:1604.03650. Retrieved from https://arxiv.org/abs/1604.03650

[3] Chan, E. R., Lin, C. Z., Chan, M. A., Nagano, K., Pan, B., De Mello, S., Gallo, O., Guibas, L., Tremblay, J., Khamis, S., Karras, T., & Wetzstein, G. (2021). Efficient Geometry-aware 3D Generative Adversarial Networks. arXiv preprint arXiv:2112.07945. Retrieved from https://arxiv.org/abs/2112.07945

[4] Schnepf, A., Vasile, F., & Tanielian, U. (2023). 3DGEN: A GAN-based approach for generating novel 3D models from image data. arXiv preprint arXiv:2312.08094. Retrieved from https://arxiv.org/pdf/2312.08094

[5] Mikeroyal (9 January 2022) MIKEROYAL/Computer-Vision-Guide: Computer Vision Guide, GitHub. Available at: https://github.com/mikeroyal/Computer-Vision-Guide (Accessed: 27 May 2024).

[6] Rahul_Roy (11 March 2024) Generative Adversarial Network (GAN). Available at: https://www.geeksforgeeks.org/generative-adversarial-network-gan/amp/ (Accessed: 27 May 2024).

[7] SAP Business Technology Platform (No date) What is data management? | definition, importance, & processes | SAP. Available at: https://www.sap.com/products/technology-platform/what-is-data-management.html (Accessed: 27 May 2024).