Documentation for the Seminar Work in Computer Vision

Author: Ludwig Jäck

1. Introduction

The goal of this project is to implement a computer vision system that can detect medical tools and track them over a sequence of frames. The requirements for the systems are to create an annotated video where the tracked objects are marked by indicators including a JSON file which stores all the position data of the objects for a given frame.

The minimum requirement is to only track the needle holder over all frames.

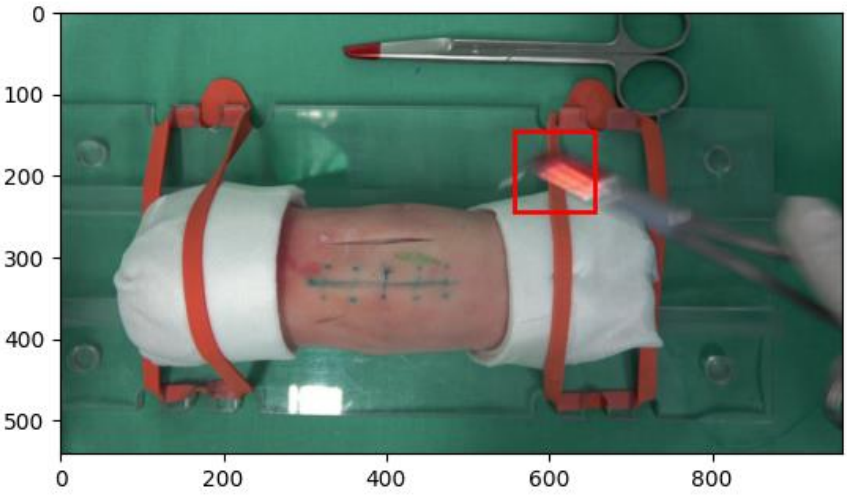

Figure 1: An example of the expected result.

The red rectangle indicates the coordinates returned by the object recognition algorithm.

2. Methods and Tools

2.1 Programming Language and Libraries

The programming language required to be used for the system is Python.
For image processing the library skimage is used.
For video encoding the library cv2 is used.
For reading the annotations data the library lxml is used.
For writing the generated annotation data the library json is used.

2.2 Color Space Analysis

For figuring out parameters for the removal of colors, a website containing HSV color space graphs was used: HSV (lukas-stratmann.com)

3. Results

3.1 Object Detection

3.1.1 Background Subtraction

By analyzing the requirements, the problem of object detection can be reduced by its two properties which describe the object. The first of those properties is that the object changes its position over time. The second property is its texture or more specific its red colored tip.

For extracting only the moving pixels in each frame, a naïve background subtraction approach is used.

$$M_{ij} = \begin{cases} 1 & if \ \left|R_{ij} - I_{ij}\right| > t_h \\ 0 & otherwise \end{cases}$$

Here M_{ij} is the mask generated by the operation above, where R_{ij} is the reference frame, which was chosen to be the first frame of every frame sequence and I_{ij} is the current frame for which the mask is getting calculated. The variable t_h here defines the thresholding after the subtraction.

The resulting mask M_{ij} is then gets applied to the frame I_{ij} so that all pixels which did not change the color between the two frames are assigned the color black.

3.1.2 Color Removal

The second part for detecting the right object now is to use the second property of the object to detect its presence in the scene. Which means there arises the need for the

removal of a lot of unnecessary colors. Therefore, the masked image is mapped to the HSV color space which enables an easier parametric removal of specific hues.

The chosen values for the color removal are finetuned in a way to only leave the needle holders tip visible in the scene after applying.

The removal includes the erase of all colors with a hue value of less than 0.9 as well as the removal of all colors with a saturation of less than 0.5.

The resulting image is then transformed back into the RGB color space where the green and blue color channels are set to zero as well.

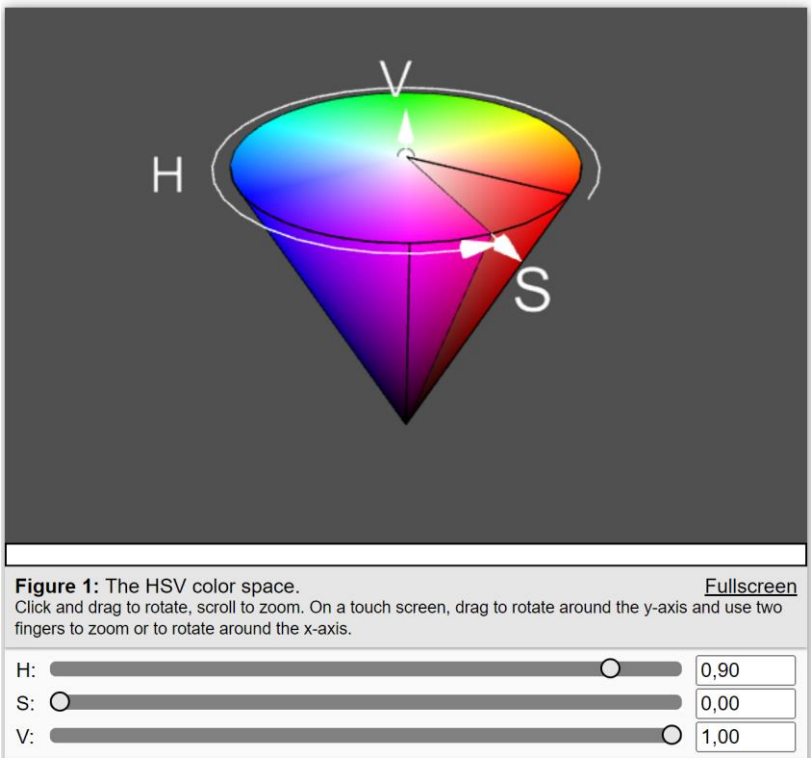

Figure 2: HSV cone used for figuring out the right color intervals

3.1.3 Object Position Identification

For calculating the position of the object for every pixel which is not black their x and y coordinates are sampled separately and for both dimensions the median value is calculated and interpreted as the most probable location of the object.

3.2 Project Structure

The project is structured into four submodules.

3.2.1 Submodule datastructs

This submodule implements two classes:

- VideoFile

  The class VideoFile serves as a collection of all frames and annotation data in memory. It implements basic getter functionality to retrieve all frames in the right order.

- Builder

  The class Builder implements the builder pattern. It is used for accumulating all the annotation data generated during the object tracking process.

3.2.2 Submodule io

This submodule implements functions for saving image frames, videos and annotation data

3.2.3 Submodule logging

This submodule implements console and visual logging available during the object tracking process.

3.2.4 Submodule preprocessing

This submodules implements all functions used for image manipulation

3.3 Running the project

The main entry point of the project is the main.py file. As the first argument it expects the path to the directory containing the annotations.xml file and the images directory.

The begin and end of the tracking process can be influences by setting the –begin and –end flags followed by the number of the respective frame.

Additional the –verbose flag can be set to level 1 or 2 to enable console logging or visual logging.

Running the main.py file will create all frames in the directory *images*, a *out.avi* file and an *annotations.json* file