Why not same result every time

Cameras – light conditions, time of day

Refer to specification and make sure all outcomes are tested

Not quantitative -> make quantitative

Robot should be quicker than a person scanning and flipping pages

How many trials is meaningful

Pareto analysis

80% of issues from 20% of system

X axis cause of failure

Y axis frequency

Accuracy vs precision

One page at a time – precision – repeatability

Motor control – accuracy test

Accuracy = num of correct % total num of predictions

|  |  |  |  |
| --- | --- | --- | --- |
| Cause of Fail | Start – 8 times | Mid – 6 times | End – 6 times |
| Silicon | 5 | 19 | 7 |
| Image Processing | 1 | 2 | 2 |
| Flicker | 2 | 6 | 3 |
| Prongs | 0 | 4 | 1 |
| Camera Unfocused |  |  |  |
|  |  |  |  |
|  |  |  |  |

Average time 160 seconds

Precision = TP / P(TP and FP)

Recoil = TP / TP+FN

Which matters more FP or FN

Not skewed – mean

Skewed – median

Categorical – mode

Say sample size – make unbiased

As described in the project plan, the functionality is prioritised using a three-tier table, with the tiers and objectives being organised using the MoSCoW model. The requirements are categorised as ‘essential’, ‘would like’ and ‘optional’ functionality. The testing and analysis is being executed in the following subsystems; the hardware, the web application and the image processing. At a later stage in the project, the tests will be integrated and then system level.

The first aspect of testing will be of the automated part of the project, the page flipper. There are a number of different requirements which ensure that the robot successfully flips pages, without tearing the paper,

Ability to correctly flip A4 notebook paper pages so the user can now scan multiple pages autonomously. We will measure the success of this feature by checking both the soft robotic finger and flicker operate correctly to flip a page. The first criteria that the page flipper will have to fulfil is that it should be able to turn 200 single pages successfully without ripping or causing any harm to them. Typically, the most pages that a notebook would have is 400, therefore we have decided that if 200 pages can be flipped autonomously then we can say that it is successful. The number 200 was decided as it correlates to the average number of pages in a notebook \cite{pages}.

The second feature that we will be testing for is that the page flipper can effectively turn pages for multiple different sizes of notebooks. The robot must have the ability to flip pages of notebooks which are A4 and A5. We will have successfully fulfilled this requirement when the robot has flipped 20 pages at a time, two runs per book, of two A4 books of different thickness and two A5 books of different thickness.

We will also test that the robot can flip pages of different materials, such as regular lined paper, glossy pages such as from magazines and printed notebooks. A successful number of trials will be that 20 single pages of each material can be flipped, twice. Compatibility with plastic paper wallets, and folders is highly desirable.

The image processing will be the second feature of the product which will be tested. The photo taken of the notebook, using a HD 720p camera, will be tested such that the photo can be processed into a scan. To ensure that the software is consistently successful, there are several test cases to consider as there are many variables which cannot be controlled. Firstly, some variables which will not be consistent are light conditions, page size, page colour, etc. For this reason, the variety in test cases of the image processing will have to be extensive, such that we can say that that the image will be converted into a scan successfully. We will ensure that the image processing will pass with a 80% success rate, with the criteria being:

1. The image text is readable.
2. The scan is of a single page.
3. Any contours in the photo are smoothed out.
4. The colour integrity is maintained.
5. There must be no excess white space surrounding the page.

These being successful will be determined by the team as they are based on perception. However, during the creation of the software, we will carry out unit testing with edge cases to ensure that the image processing is effective.

The variation of test cases will include:

1. Images taken in bright, medium and low lighting.
2. Images with different sizes of font and handwriting.
3. Images with different background colour of pages, not only white but darker coloured.
4. Images which have paper of different materials, as described in the testing of the page flipper.
5. We will also test the scanning with different coloured surroundings, which would influence the design of the page flipper.

The second part of the software which will be tested is the webpage functionalities. We will carry out unit testing to ensure that a user can create an account, with a verified password, login to and logout of the website. There will be further unit tests which ensure that a user can add, delete and view folders, in which scans can saved into. Users must be able to upload scans into chosen folders and delete existing scan, furthermore users should be able to download and export the scans onto their own device. Unit testing will be an important way to isolate any issues with the code and solve them efficiently. The testing will be successful if the users are able to complete all of these desired outcomes.

To ensure that the website is compatible with other devices, we will test that the functionality is maintained when being used on different devices, browsers and operating systems.

\* Overall what test works and what didnt-> how we have to adjust within timeline to reach clear success criteria goals \*

Image Processing

To turn a photo of a notebook page into a scanned image using a Python file, the first step is to load the image, then the image is pre-processed to reduce noise and convert it to a binary image, which makes it easier to detect the edges of the page. Edge detection algorithms are then used to detect the edges of the page contour in the image. The background of the image is masked using the detected edges of the page contour, which is then used to crop the image and remove the background. Finally, a perspective transformation is applied to the cropped image to straighten it and make it a scanned image. The binary image can be converted back to a colour image by assigning a colour to each of the two values in the binary image and creating a colour map that maps the binary values to the desired colours. The transformed image is then saved in any desired format, such as JPG or PNG.

1. Load the image
2. Pre-process the image

* reduce noise
* convert it to a binary image

1. Use edge detection algorithms such as to detect the edges of the page contour in the image
2. Mask the background of the image using the detected edges of the page contour
3. Crop the image using the masked background
4. Apply a perspective transformation to the cropped image to straighten it and make it a scanned image
5. Convert the binary image back to a colour image by creating a colour map that maps the binary values to the desired colours.
6. Save the transformed image in any desired format, such as JPG or PNG
7. algorithm for image processing

Diagram

Description automatically generated

https://www.researchgate.net/publication/340556042\_Image\_recognition\_and\_diagnosis\_for\_vibration\_characteristics\_of\_cone\_valve\_core