Microbeads

Josiah Carpenter, McKenna Gates, Luke Johnson, Jacob Wakefield, William Zay, Noah Zeilmann

September 9th, 2018

Submitted to
Dr. Razib Iqbal
Missouri State University
CSC 450 Course Project
Fall 2018

Project Title: Microbeads		
Team Information	1. Josiah Carpenter 2. McKenna Gates 3. Luke Johnson 4. Jacob Wakefield 5. William Zay 6. Noah Zeilmann	
Document Type: Software Requirement Specification v1.0		*Last updated on*: 09-09-2018

Document Approval

This Software Requirements Specification has been accepted and approved by the following stakeholders:

Printed Name	**Title**	**Signature and Date**
K. YOSHIMATSU	Customer	
R. IQBAL	Customer/Course Instructor	

Table of Contents

1. Introduction

The purpose of the microbead software is to allow researchers to utilize microbead dye technology with relatively affordable equipment. This will be achieved by creating software to analyze images of slides from a microscope to determine the number of beads and colors in the slide. The software will detect overlap in images to prevent beads from being double counted and return a stitched image to the user. The introduction section gives a clear and concise description of the microbead software's purpose, scope, common abbreviations, and references used in this software and throughout the SRS documentation.

1.1 Purpose

This SRS is intended to provide a detailed outline of functionality, requirements and dependencies, and other specifics for the microbead software. It is intended for current and future developers as well as the stakeholders in the project, Dr. Razib Iqbal and Dr. Keiichi Yoshimatsu, to make sure the developers are keeping the project in the proper scope and developing accurate functionality.

1.2 Scope

The application will be given a set of pictures of microbeads under a microscope and correctly identify beads. The software will count these beads and return the average color intensity of the microbeads, in RGB and Grayscale, to the user. The user will also receive a map of the beads made by stitching together the pictures it is given. The software will not count broken beads, overlapping beads, or drops of water. This will provide users a cost effective way to obtain data on the beads without doing it by hand.

1.3 Definitions, Acronyms, and Abbreviations

CV - Computer vision. A subfield of computer science used to extract data from images and videos.
GUI - Graphical user interface. The user interface for interacting with a software system.
JSON - JavaScript Object Notation. A lightweight data-interchange format
Microbead - A plastic bead less than a millimeter in diameter.
SIFT - Scale-invariant feature transform. This is an algorithm used for object recognition.
Stitching - Merging multiple images into a single image.

1.4 References

1. A Liquid Array Platform for the Multiplexed Analysis of Synthetic Molecule–Protein Interactions
Todd M. Doran and Thomas Kodadek
ACS Chemical Biology 2014 *9* (2), 339-346
DOI: 10.1021/cb400806r

1.5 Overview

This SRS will be broken up by description, requirements, and change management process.

Each of these sections will be broken up further and go into specifics. The description will include product functions, constraints, assumptions, and dependencies. The requirements section will break down the requirements into functional and non-functional, design constraints, and any other specifics about how the software will work. Finally, the change management section will go over how the SRS will be updated when the scope or functionality changes by either the developers or the customers.

2. General Description

This segment of the SRS will give an overview of the software as a whole. The software in its entirety will be explained to show how it will interact with the data from the images it processes. It will also describe how the user will interact with the software and what can be expected after they input images.

2.1 Product Perspective

This software is based off of research done by the Departments of Chemistry and Cancer Biology at The Scripps Research Institute in Florida. This research determines color intensity of protein beads that have been colored using a special dye that only binds to certain proteins to aid in disease detection. Our software aims at producing a cost-effective solution to this process. The software will also act as a foundation for future research involving proteins and disease-markers.

2.2 Product Functions

The software will take a set of pictures containing microbeads and construct a map of the beads by stitching the pictures together to create one larger image. It will do this by detecting overlap in the photos and put the pictures where they belong in the map. The software will also give the user information about the set of microbeads containing the number of the beads of each color as well as the average color intensity in RGB and Grayscale.

2.3 User Characteristics

The intended users will be anyone given authority to do so by Dr. Yoshimatsu. The user will be expected to have access to the necessary materials provided by the chemistry lab in order to obtain a set of images of microbeads with dye under a microscope.

2.4 General Constraints

Considering that the microbead software will be used for research, it must count the beads to a high degree of accuracy.

Another constraint is the deadline of the project. The final demo of the project is 12/08/2018, however, we plan to complete the project at most three weeks before that date.

2.5 Assumptions and Dependencies

The targeted operating system is Windows. In order for the system to run, the Microbeads software and Python 3.6 must be on the client's machine. The client will submit images obtained using a microscope. The quality of the images should be sufficient for accurate processing. If the user does not submit images of microbeads of sufficient quality, the accuracy of the software could be negatively affected and may give the user invalid results.3. Specific Requirements

3. Specific Requirements

3.1 Functional Requirements

3.1.1 The web server will launch on execution.

Source: Team decision
Priority: TBD
Introduction: On execution of the main Python script, the web server will deploy in the background.
Inputs: None
Processing: None
Outputs: Web server
Error Handling: None

3.1.2 The web client shall accept multiple images to be sent to the web server.

Source: Customer meeting
Priority: TBD
Introduction: These images shall be selected using a file browser and will be sent to the web server for stitching.
Inputs: Multiple images from client.
Processing: The images will be sent from the web client to the web server, then be processed by the stitching algorithm.
Outputs: Multiple images on the server.
Error Handling: To handle the error of failing to send multiple images to the web server, the user will be prompted with a notification on the web client.

3.1.3 The web client shall accept a video file to be sent to the web server.

Source: Customer meeting
Priority: TBD
Introduction: The video will be selected using a file browser and will be sent to the web server for stitching.
Inputs: Video file
Processing: The video file will be sent from the web client to the web server to be processed by the algorithm for stitching.
Outputs: Status code from the server.
Error Handling: To handle the error of failing to send a video file to the web server, the user will be prompted with a notification on the web client.

3.1.4 The web server shall stitch the multiple input images into one single image.

Source: Project description
Priority: TBD
Introduction: The web server receives images from the web client and uses OpenCV to stitch the images into one image for the user.

Inputs: Multiple images
Processing: Multiple images are passed to image stitching algorithm to be processed.
Outputs: Stitched image
Error Handling: None

3.1.5 The web server shall create a single panoramic photo from the input video file.
Source: Project description
Priority: TBD
Introduction: The web server shall use OpenCV to create a single image that will be sent to the user as well as to the SIFT algorithm to count the beads.
Inputs: Video file
Processing: Multiple images are passed to image stitching algorithm to be processed.
Outputs: Panoramic image
Error Handling: None

3.1.6 The microbead software shall detect and remove overlap in the input images while stitching.
Source: Project description
Priority: TBD
Introduction: The stitching process shall create a single image of the entire slide.
Inputs: A collection of microbead slide images
Processing: Using the stitching algorithm from OpenCV, the software will determine which pictures have overlapping sections by comparing each image. Those images will then be combined.
Outputs: A collection of images without overlap.
Error Handling: None

3.1.7 The microbead software shall count the amount of beads present in the stitched image.
Source: Project description
Priority: TBD
Introduction: The software shall count beads in the stitched image that it generates and return that information to the user.
Inputs: Stitched image of microbeads
Processing: Using SIFT to generate keypoints, each keypoint will be analyzed to determine the existence of, and count, valid beads.
Outputs: JSON data, CSV file
Error Handling: None

3.1.8 The microbead software shall determine the color intensity of each bead in the stitched image.
Source: Project description
Priority: TBD
Introduction: The software shall gather the color intensity of each bead so that it can be returned to the user.

Inputs: Stitched image of micro-beads.
Processing: Using SIFT to generate keypoints, each keypoint will be analyzed to determine the existence of, and count, valid beads.
Outputs: JSON Data, CSV file
Error Handling: None

3.1.9 The microbead software shall not count water droplets in the stitched image.
Source: Customer meeting
Priority: TBD
Introduction: In order to maintain an accurate count of beads the software must not mistakenly identify water droplets as beads.
Inputs: A stitched image of microbeads.
Processing: Using SIFT to generate keypoints, each keypoint will be analyzed to determine if it is a valid bead or a water droplet.
Outputs: None
Error Handling: None

3.1.10 The microbead software shall not count crushed beads in the stitched image.
Source: Project description
Priority: TBD
Introduction: The software needs to be accurate and therefore it should not include water drops in the count of microbeads.
Inputs: A stitched image of microbeads.
Processing: Using SIFT to generate keypoints, each keypoint will be analyzed to determine if the bead is valid or crushed.
Outputs: None
Error Handling: None

3.1.11 The web server shall pass analysis results to web client.
Source: Project description
Priority: TBD
Introduction: The GUI shall receive the results and present them to the user.
Inputs: JSON data
Processing: The data gathered from the bead analysis will be sent from the server to the client.
Outputs: JSON data, CSV file
Error Handling: None

3.2 Non-Functional Requirements

3.2.1 The images shall be sent from the client to the server with a POST request.
Source: Team Decision
Priority: TBD
Introduction: Image processing requires the images to be sent to the server through an HTTP POST request.

Inputs: A collection of .jpg images of microbeads from the client
Processing: After specifying the images to upload and putting them in the correct order, the client will click a ‘submit’ button which will initiate the HTTP POST request.
Outputs: A data structure containing the .jpg images from the client ready to be manipulated by the stitching algorithm.
Error Handling: The server should respond with a successful status code. If the server responds with an error code the client will be notified that the request is not successful.

3.2.2 The MP4 video shall be sent from the client to the server with a POST request.

Source: Team Decision
Priority: TBD
Introduction: MP4 video processing requires the MP4 video to be sent to the server through an HTTP POST request.
Inputs: An MP4 video of the microbeads slide.
Processing: After specifying the MP4 video to upload, the client will click a ‘submit’ button which will initiate the HTTP POST request.
Outputs: A data structure containing the MP4 video from the client ready to be manipulated by a counting algorithm.
Error Handling: The server should respond with a successful status code. If the server responds with an error code the client will be notified that the request is not successful.

3.2.3 The microbead software shall use the Flask Web Framework.

Source: Team Decision
Priority: TBD
Introduction: Flask will allow the software to create a local web server which will host the GUI.
Inputs: None
Processing: None
Outputs: None
Error Handling: None

3.2.4 The microbead software shall use the SIFT to analyze the image.

Source: Client Suggestion / Team Decision
Priority: TBD
Introduction: SIFT will detect features within the image that will be used to identify beads.
Inputs: Stitched image of user provided images
Processing: The SIFT algorithm with be applied to the stitched image to find all circles.
Outputs: Data about the analysis and keypoints in the image.
Error Handling:

3.2.5 The microbead software shall use Numpy.

Source: Team
Priority: TBD

Introduction: This library is required because Python lacks native support for the arrays required to manipulate images.
Inputs: None
Processing: Numpy will be used by OpenCV for array functionality.
Outputs: None
Error Handling: None

3.2.6 The microbead software shall use OpenCV to stitch images together.

Source: Team decision
Priority: TBD
Introduction: OpenCV will detect overlap in images and stitch them together to prevent beads that appear in more than one image from being counted more than once.
Inputs: None
Processing: None
Outputs: None
Error Handling: None

3.2.7 The microbead software shall return stitched images for user verification.

Source: Team decision
Priority: TBD
Introduction: Since accuracy is critical, the user will have the ability to verify that the images have been successfully stitched together before the counting process begins.
Inputs: Stitched image
Processing: The user will be presented with a stitched image, in which they can either approve or reject the stitched image.
Outputs: Verified stitched image
Error Handling: None

3.2.8 The GUI shall have a user defined number of rows and columns for arranging images.

Source: Team decision
Priority: TBD
Introduction: In order to aid in the stitching of images the user shall define a number of rows and columns in which the they can arrange images in approximate order.
Inputs: A number of rows and columns.
Processing: A grid will be dynamically created based on the number of rows and columns specified by the client using JavaScript.
Outputs: A grid with enough cells for the client to populate with images.
Error Handling: None

3.3 Design Constraints

3.3.1 Export data shall be in CSV format.

Source: Team decision
Priority: TBD
Introduction: The customer has indicated that CSV is an acceptable format for the export data.
Inputs: A JSON object containing data about the microbeads.
Processing: Analyze the JSON object and translate to CSV.
Outputs: CSV file with correct data
Error Handling: None

3.3.2 Input Images shall be in JPG format.

Source: Team decision
Priority: TBD
Introduction: Since JPG is the default format for the microscope camera, the system will accept JPG images.
Inputs: None
Processing: None
Outputs: None
Error Handling: If the system is provided with images in a format other than JPG the user will be notified and the images will not be processed.

3.3.3 Input Videos shall be in MP4 format.

Source: Team decision
Priority: TBD
Introduction: MP4 is a standard video format that the microscope camera uses. Therefore the system will accept MP4 format for videos.
Inputs: None
Processing: None
Outputs: None
Error Handling: If the system is provided with a video format other than MP4 the user will be notified and the video will not be processed.

3.4 External Interface Requirements

3.4.1 User Interfaces

3.4.1.1 The microbead software shall have a web based GUI.

Source: Team Decision
Priority: TBD

Introduction: Having a graphical interface will allow the software to be used by non-technical users.
Inputs: None
Processing: None
Outputs: None
Error Handling: None

3.4.2 Hardware Interfaces

Not Applicable

3.4.3 Software Interfaces

Not Applicable

3.4.4 Communications Interfaces

Not Applicable

3.5 Logical Database Requirements

Not Applicable

3.6 Other Requirements

All software used shall be open-source so researchers will have open access to it.

4. Change Management Process

All changes made to the Microbead software will fall into on of the following two categories. These categories will provide definitions as to how a change can be proposed, validated, and presented to the stakeholders for approval.

1: Developer Change - A developer sees a change that could improve or expand the functionality of the software.

- Team Contact: The developer will contact the entire group by either Slack or email. The developer should include a brief, clear statement on what needs to change and a short justification for the change.
- Meeting: At least three of the six team members shall meet in person to discuss the change and its effects on the current system in detail. The group that meets will determine the validity of the change and if it meets the scope of the project.
- Team Vote: To determine if the change should be presented to the user, there will be a team vote after the in person meeting. For the change to be considered valid and move on to the stakeholder it must receive three votes from the six team members.
- Proposal Document: A professional document describing the change will be written to present to the stakeholders. The document will include a clear statement describing the proposed change and any additional information needed to detail the change. The document should also include the justification for the change.
- Present to Stakeholder: The group will meet in person with the stakeholders to present the proposal document of the change to be made. The group and the stakeholders will then decide if the change needs to be implemented.

2: Stakeholder Change - A stakeholder deems a change necessary to improve or expand the functionality of the software.

- Meeting: The stakeholders and the group will meet to discuss a change that the stakeholder deems necessary. The group will gather the detail needed to fully understand the change.
- Proposal Document: The group will create a professional document stating the description of the change and the reason for it.
- Present to Stakeholder: The proposal document will either be presented to the stakeholders in person or over email to ensure that the change has been understood correctly.