Robust Three-Dimensional Facial Reconstruction with Occlusions

Elise Donszelmann-Lund, Isabel Frolick, Sophie Ellwood

December 2024

To access the project GitHub repository, please visit: https://github.com/izzyfrolick/COMP558_FacialReconstruction. This repository will be kept publicly available until 12/31/2024. Data provided on request.

Abstract

Facial reconstruction is a challenging task, especially in the presence of visual occlusions such as face masks or sunglasses. While existing methods achieve high accuracy, they often fail to generalize occluded scenarios where features are obscured at varying angles or scales and are too computationally expensive for real-time use.

This paper proposes a novel method to perform accurate, fast facial reconstruction under occluded conditions using RGB-depth images. Our approach integrates non-occluded reference images for registration, allowing for the reconstruction of the occluded region itself. Using the depth data from an RGB-depth camera, our method registers non-occluded and occluded images, segments the occlusion, and reconstructs images with four tested techniques.

The proposed pipelines were tested with various transformations, rotations and occlusion types. Two specific scenarios (x-translation with sunglasses occlusion and no-movement with mask occlusion) were discussed. The naive approach consisting of a 2D affine transformation, k-means clustering and morphological thresholding was found to be the most effective.

1 Introduction

This paper moves to implement a method using classical image reconstruction and computer vision techniques to execute accurate facial reconstruction on RGB-depth images when facial occlusions are present.

1.1 Motivation

Image reconstruction is a general problem encompassing many applications in which an object is estimated from a degraded image of the object. Image reconstruction becomes hard (among other instances) when visual occlusions are present. Visual occlusions are a complete or partial blockage or obscured view of an object which cause performance degradation when attempting to reconstruct the object.

Using current methods, to be discussed in Section 1.2, it is generally feasible to reconstruct coarse-grain features with a high-degree of accuracy, especially when a non-occluded, reference image is attained. However, image reconstruction becomes more difficult when the features are fine grain, such as facial reconstruction, where the goal is to construct a reconstructed image that sufficiently resembles the individual. Facial reconstruction is difficult when occlusions are present, such as a face mask, sunglasses, or other synthetic occlusions, due to the precision required and fine scale of the features. Current methods, as will be discussed in 2, struggle to perform well on facial reconstruction when occlusions are present.

This difficult problem is key in applications such as biometric authentication, forensic reconstruction, and surveillance recognition where it is rare to have unobstructed facial data of the person of interest. In these applications, it is necessary to extract distinctive and correct facial information to avoid false identification, as it could lead to incorrect judicial prosecution or false identification of a deceased person.

1.2 Related Work

While inpainting methods such as MICE can work well for small occlusions [1], more sophisticated models are needed when large parts of the face are blocked. Although using 2.5D and 3D data is often used in facial reconstruction, the acquisition time for this data can be high, and impractical for real-time systems. Depth cameras like the Kinect have been used [2] to quickly synthesize RGB and depth data. An issue for these methods is the lack of sufficient datasets which are conditional upon having seen both many similar faces, and the occluding object before. While traditional facial detection has focused on methods like PCA [3] deep learning methods have become popular for facial reconstruction [4]. Many models perform well on traditional facial reconstruction tasks but struggle when occlusion is introduced. As previously shown, training data taken at unobstructed facial views can make models perform poorly when the occlusion occurs when the face is at an angle [5]. Even with improved deep learning models, the quality of occlusion detection has a large impact on the facial reconstruction.

1.3 Proposed Solution

A non-occluded reference image will be used for registration and to provide the non-occluded visual form of the occluded region. Using a reference image is a non-trivial solution as many of the applications relevant to this problem will have an exhaustive database, such as police surveillance recognition through the federal drivers' license database or hospitals identifying masked employees to enter the operating room.

The facial occlusions that will be tested include a blue medical face mask, black sunglasses, pink sunglasses, scarf, and hands shielding the face. The primary occlusions considered will be the face mask and the black sunglasses as they each occlude a distinct region of the face, but do not deform the face and are distinct colours against the other features of the subject.

A naïve case will first be implemented to test the functionality of the pipeline. In this naïve case, all movement of rotation and translation will be minimized such that registration will be simple and the algorithmic focus will be on segmenting the occluded region. Once the

naïve case is completed, further data acquisition will include harder scenarios with rotations and translations including x-direction translation, z-direction translation (subject moves toward the camera), z-axis rotation of the head, y-axis rotation of the head (into profile), z-axis rotation of the torso.

The proposed algorithm is as follows:

1. Occluded and non-occluded images are collected for each scenario using a PrimeSense Carmine 1.09 RGB-depth camera.
2. The video streams from the depth and colour images are converted to numpy arrays. Four corresponding frames are used: colour and depth frames for the non-occluded image and the depth and colour frames for the non-occluded image.
3. Register the images using the depth information from the PrimeSense camera using Iterative Closest Points or similar. This will transform the non-occluded image into the same space as the occluded image.
4. Segment the region where the optical RGB values are different between the transformed non-occluded image and the occluded image. This will give only the occlusion itself.
5. Create a binary mask of the occluded area.
6. Project the occlusion mask onto the non-occluded image to create a non-occluded mask, project the non-occluded mask onto the occluded image for facial reconstruction.

2 Methods

In this section, both the methods that were used in the final pipeline, and those that were implemented but not included in the final pipeline to reconstruct the occluded regions will be discussed.

2.1 Data Acquisition

Data was collected using a PrimeSense Carmine 0.9 RGB-D camera. The subject was captured with and without an occlusion for each scenario. Different scenarios included no translation, x,y,z-translations, and z-axis rotations. Occlusions used included a mask, scarf, gloves and sunglasses. Example data acquisitions are shown below in Table 1.

2.2 Data Processing

Once the data and its variations had been acquired, it was necessary to process the images so that the usability and flexibility of the pipelines were maintained. To process the data, the backgrounds were removed and the RGB image were converted to hue-saturation-value (HSV) images.

2.2.1 Image Manipulation

In the scenarios tested, regardless of rotation or translation, each acquisition consisted of a single subject against a plain white background, which constituted much of the image. As the background was a planar surface, it was trivial to threshold the depth images such that any pixels outside the range of the subject's depth were set to zero. In many of the scenarios, the range that was used to mask the background pixels could be kept constant - all pixels outside a depth from 1000-1380mm were removed. However, this thresholding is not an elegant solution and does require manually changing the masking thresholds, decreasing pipeline flexibility. Overall, removing the background is a necessary step and leads to better key point detection, transformation, and segmentation.

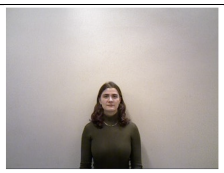
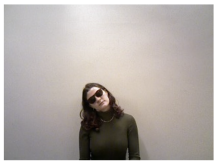
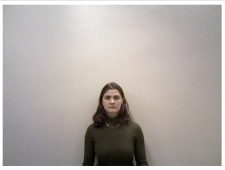
	Non-Occluded Image	Occluded Image
Z-axis Head Rotation (RGB Images)		
X-Translation (RGB Images)		
Z-Translation (Depth Images)		

Table 1: Examples of Collected Data

Figure 1: Comparison of Preprocessed Image (Left) and Depth Image with Removed Background (Right) [Purple is Depth=0]

2.2.2 Colour Space Manipulation

At the end of the pipeline, it would be necessary to have a mask of the occluded area such the nonoccluded face could be projected back onto the occluded area. As such, it was necessary to get the pixel-wise RGB difference between the occluded image and the transformed nonoccluded image. Ideally, the difference image would be the occluded object against a purely black (0,0,0) background, however, due to noise and imperfections, some of the subject was still visible so segmentation on the occluded object was not functional.

To solve this, the RGB image was converted to a HSV (hue-saturation-value) image to increase the difference in the colour values between the occluded region and the subject. The value channel (brightness) produced the best results when compared to the RGB difference image, all three HSV channels, the hue channel, and the saturation channel.

2.3 Feature Detection & Matching Algorithms

To register the non-occluded and the occluded RGB-d images, it is necessary to find corresponding points between the images. Although finding corresponding keypoints between the images is not necessary for the Iterative Closest Points (ICP) method, which transforms all points between two point clouds for registration [6]. It was decided to implement Affine Transformation

Figure 2: Example of the difference image between the two registered images, and its corresponding single-channel (Value) HSV image

to test the depth image registration, rather than point cloud registration as its more computationally efficient. To use an Affine Transformation on the depth images, it was necessary to find corresponding keypoints by implementing feature detection and feature matching on the images. Three methods were attempted: ORB feature detection with a Hamming Distance metric for matching, SIFT feature detection with a FLANN-based metric for matching, and a mean squared error method for feature detection. In the final pipeline, the ORB feature detection with a Hamming Distance metric was implemented.

2.3.1 SIFT Feature Detection

The Scale-Invariant Feature Transform (SIFT) algorithm [7], specifically the OpenCV implementation, was tested as a key point detector for normalized depth images. The algorithm parameters used were a contrast threshold of 0.006 and an edge threshold of 15.

SIFT identifies key points by searching for local extrema in the scale-space representation of an image. The scale-space is constructed using a Difference of Gaussian (DoG) pyramid, defined as:

$$\mathrm{DoG}(x, y, \sigma) = G(x, y, k\sigma) - G(x, y, \sigma)$$

where $G(x, y, \sigma)$ is a Gaussian-blurred image at scale σ, and k is a scaling factor. Extrema are identified by comparing neighborhood pixels in both spatial and scale spaces. Key point localization and orientation assignment refines points and ensures rotation invariance by computing gradient orientations in the key point's neighborhood.

Feature matching between the non-occluded and occluded depth images was computed using a Fast Library for Approximate Nearest Neighbors [8] (FLANN) matcher from OpenCV. FLANN finds the nearest neighbors of each descriptor using a KD-Tree-based search. A two-nearest-neighbor search was used for each descriptor, where similarity was computed using the Euclidean distance between two descriptors:

$$d(\mathbf{p}_i, \mathbf{p}_j) = \sqrt{\sum_{k=1}^{N} (p_{i,k} - p_{j,k})^2}$$

where $\mathbf{p}_i$ and $\mathbf{p}_j$ are two descriptor vectors, and N is the number of dimensions.

The ratio between the best match and the second-best match was computed. Matches were kept only if this ratio was less than 60%. Specifically, key point matches were retained only if:

$$\frac{d_{\text{best}}}{d_{\text{second}}} < 0.6$$

where d_{best} and d_{second} are the Euclidean distances to the best and second-best matches, respectively.

2.3.2 ORB Feature Detection

To detect and match the corresponding features between the depth images, the Oriented FAST and rBRIEF (ORB) algorithm was used [9]. Although not covered in class, ORB operates very similarly to SIFT, detecting keypoints in each scale of the input Gaussian pyramid. Compared to SIFT, ORB is simple and computationally efficient, but it lacks the robustness to detect features that change in scale and angle. Given the simplicity of the images that were used, this was not a problem on this pipeline. In this instance, a built-in ORB detector from the OpenCV library was used with a maximum of 250 features to detect in each image. To compute feature matching, a brute-force algorithm was implemented using a Hamming distance metric [10]. The suitable corresponding keypoints found were sorted, based on the distance metric, and the mean average of the distances was calculated and used as a dynamic threshold, such that for any scenario run, the ORB keypoints kept will always be equal to or below the mean Hamming distance.

The ORB feature detection worked the best of any algorithm implemented, with low Hamming distance error and a high number of keypoints. This was the implementation that was executed in the pipeline.

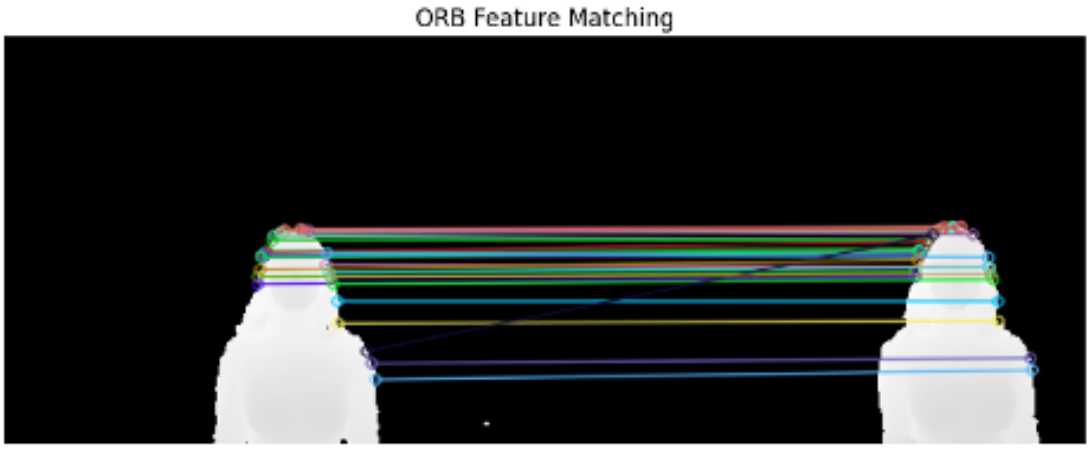


Figure 3: Example of corresponding keypoints found using ORB and Hamming Distance metric algorithm, on the x-translation sunglasses case

2.3.3 Mean Squared Error Neighbourhood Feature Detection

Another simple approach to matching features was to use mean squared error (MSE). By iterating over all pixels, a window around that region was taken in both the occluding and non-occluding depth images. By using MSE as a similarity measure, matches were identified based on having a score lower than a pre-set threshold. While making sense for the naive case, this method did not work when transformations were introduced and therefore SIFT and ORB were the primary means of feature detection.

2.4 Image Registration Algorithms

To reconcile the original image with the occluded image, a transformation was found between the two. In any case other than the naive one, the subjects in the images were not aligned. Examples of movements included translations and rotations about multiple axes, meaning finding the pixel-wise difference between the occluded and non-occluded images would not work. Two methods: affine transformations and ICP were implemented to align the subjects, and so 'overlap' them in the same space. While affine transformations relied on feature detection done in previous steps, ICP used point clouds.

2.4.1 Two-dimensional Rigid Affine Transformation

After finding corresponding keypoints across the two images, the occluded and non-occluded images, one method that was attempted to compute the transformation between the images was a 2-dimensional affine transformation [11]. To optimize the affine transformation, a custom transformation using least squares and optimized through iterations of RANSAC was used [12]. First, three points from the corresponding keypoints, the minimum number required for affine, were randomly selected, and the least squares solution was found. The transformation was then applied to all points and the error was computed as the Euclidean distance between the transformed and the actual points.

To ensure the best affine transformation was used, this custom method was compared to the built-in CV2 estimateAffine2D function, which uses the cv2 library RANSAC method to compute the transformation.

Figure 4: Comparison of Custom Affine Transformation Function (Left) and OpenCV Built-in Affine Transformation (Right)

These 2D affine transformation methods were compared in the scenario with X-axis translation and the parameters were set to the same values; a minimum consensus size of 10.0 was required, the maximum number of iterations was set to 2000, confidence was kept at 0.99, the number of refined iterations was kept at its default, 10.

Table 2: Comparison of Custom 2D Affine and Built-In Affine Transformations.

	Custom Transformation	**Built-In Transformation**
Inliers Count	48	30
Mean Residual Error	8.6	24.2
Overlapping Pixels Count	8430	8076

The custom 2D affine transformation made improvements over the built-in method and was therefore used in the pipeline.

2.4.2 ICP Registration

Alignment between the non-occluded and occluded depth value point clouds was performed using the Iterative Closest Point (ICP) algorithm from the Open3D library. This algorithm iteratively minimizes the Euclidean distance between corresponding points between a source and target point cloud and estimates rigid transformations between the two point clouds [13].

An identity matrix was used as an initial transformation matrix and a distance threshold of 80 pixels was chosen as the maximum distance allowed between points; a default value of 30 maximum iterations was used. A correspondence set from target point cloud (occlusion depth values), $\mathcal{P}$, and the source point cloud (non-occluded depth values), $\mathcal{Q}$, was identified, transforming Q using the current transformation matrix T. The transformation matrix, $\mathcal{T}$, was updated by minimizing an objective function $\mathcal{E}(\mathcal{T})$, based on the correspondence set.

$$E(T) = \sum_{(p,q)} \|\mathbf{p} - \mathbf{Tq}\|^2 \tag{1}$$

A fitness score, representing the number of inliers relative to the total number of points in the source point cloud, and the root mean square error (RMSE) between the matched points, were used to evaluate ICP performance. The ICP images were then converted back to 2D depth images using the intrinsic information of the camera.

2.5 Occlusion Segmentation Methods

Next, the object occluding a face was identified through segmentation. Two different clustering algorithms, K-Means and DBSCAN were used to generate a binary mask of the object. These unsupervised clustering algorithms were chosen for their flexibility in identifying distinct objects in a range of settings. The models were applied to images containing masks, sunglasses, scarves, and gloves obstructing the face.

2.5.1 K-means Clustering

To segment the occlusion from the image, k-means clustering on the occluded image's brightness channel from the HSV image was implemented. K-means from the openCV library was used, with a stopping criterion of 0.2 center movement, 30 maximum iterations and $k = 2$.

K-means clustering [14] is an unsupervised learning algorithm that partitions k clusters by minimizing within-cluster variance. Cluster centers are initialized as (u, j), and points are assigned to the cluster whose center is nearest in an iterative manner.

$$\mathbf{c}_i = \arg\min_j \|\mathbf{x}_i - \mathbf{u}_j\|^2 \tag{2}$$

Cluster centers are updated by computing the mean of all points assigned to it. This is repeated until the movement of cluster centers between iterations is less than a defined threshold, or the maximum number of iterations is reached.

Cluster labels were mapped back to the brightness image, where ideally, the occlusion was one of the two clusters computed. To further isolate the clustered occlusion, morphological thresholding, which closes small holes and gaps within a cluster, was used.

Morphological thresholding is comprised of dilation, erosion and closing operations to fill gaps and smooth boundaries of segmented regions. OpenCV's morphological closing operation was used.

2.5.2 DBSCAN and Haar Cascade

While K-means clustering provided a way to identify simple objects in the image, a more robust model was tested to find objects with arbitrary shapes. The unsupervised algorithm Density-Based Spatial Clustering of Applications with Noise (DBSCAN) from scikit-learn was chosen to detect occluding objects such as gloves, sunglasses, and masks [15]. First, depending on the type of occlusion, the hue or saturation channel was taken from the input image, allowing for higher contrast between the profile and the object. DBSCAN was run on arrays containing both the colour and position of every pixel in the image. The hyperparameters 'epsilon' (the maximum distance for two points to be considered neighbours) and 'min samples' (the fewest number of points required to form a cluster) varied based on the setting in which the image was taken. Haar Cascade facial detection from OpenCV was used to identify the cluster that best fit the facial occluding object. This facial detection model passes rectangular kernels over positively and negatively classified faces to learn features sequentially[16]. A binary mask of the occluding object was found by selecting the cluster with the highest overlap with the face region detected by Haar Cascade.

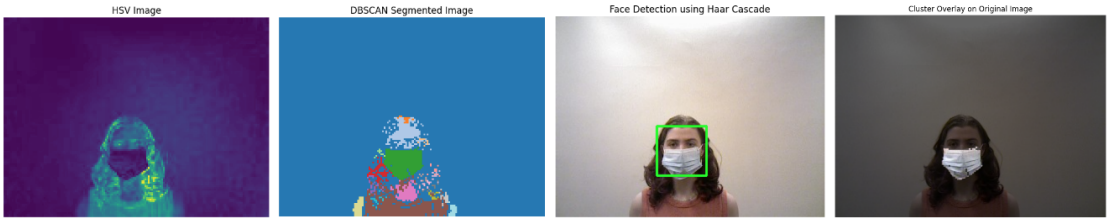


Figure 5: Clustering applied on HSV image. Object cluster is chosen as the one with highest overlap with face region.

2.6 Reconstruction

To reconstruct the face, pixels from the non-occluded image were projected onto obstructed areas in the occluded image. As a result of the transformation done in earlier steps, the two images were aligned in the same space. By using the binary mask generated through segmentation, pixels from the original image were mapped to the occluded image. The occlusion was therefore effectively removed by overwriting those select pixels while preserving the remainder of the profile and surroundings.

3 Results

This section outlines the four algorithmic pipelines implemented throughout the execution of this project: the naïve pipeline, the DBSCAN pipeline, the ICP-to-depth-image pipeline, and the ICP point cloud pipeline. When implementing the naïve pipeline, the structure of the proposed algorithm was followed in its simplest form and the following three pipelines were adapted where this algorithm failed.

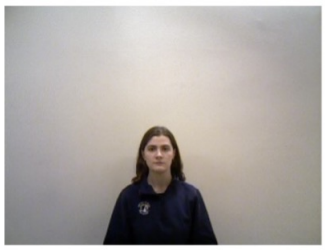
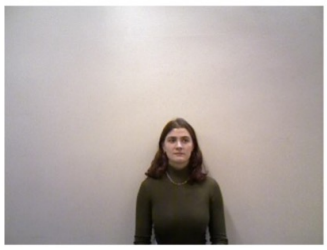
As the latter three pipelines were resultant from the first, all four pipelines follow the same first few steps. First, the images were preprocessed as summarized in Section 2.2 where the depth background pixels were masked to zero. Then, using these masked depth images, 3D point clouds were constructed using the Open3D library. Next, the corresponding features between the two depth images were found using the ORB feature detector and a Hamming distance matching metric. Although each step was not required for every pipeline, it was determined that keeping all the pipelines similar would improve the clarity and readability across group members.

After the key point matching, each pipeline differed in its implementation of the next steps: image registration and transformation, occlusion segmentation, and reconstruction, as will be outlined below.

3.1 Pipeline #1: Naïve Pipeline

After the processing steps had been completed, as outlined above in 2, the 2D affine transformation was implemented, as seen in 2.4.1, and the difference between the registered RGB images was found. The difference image was then converted into an HSV colour space and parsed to only include the value channel (brightness) as described in 2.2.2. From there, the K-means clustering algorithm (2.5.1) was implemented to derive the cluster where the occluded region occurred and the cluster underwent thresholding to derive the occlusion mask. To threshold, morphological thresholding was used for region fill in, the filled in occlusion mask underwent denoising, and median blurring was performed. Lastly, the binary mask was projected onto the occluded image, the region of the mask was used on the non-occluded image, and the non-occluded binary mask was projected onto the occluded image for reconstruction. To visualize the results of this pipeline, the naïve face mask case where no movement occurs and the sunglasses case with x-translation are below, in Table 4.

Table 3: Input images used for pipelines, shows both the occluded and non-occluded images.

Case 1: Naive Case, Face Mask	**Case 2: X-Translation, Sunglasses**
	
	

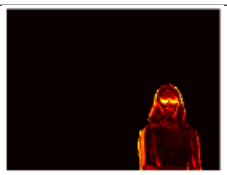
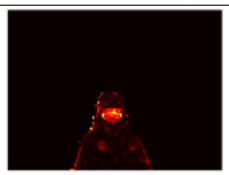
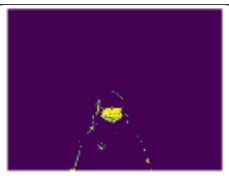
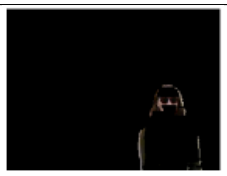
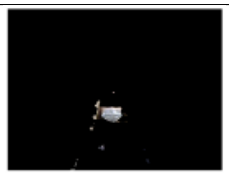
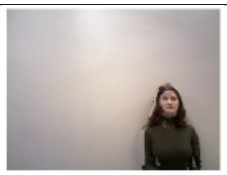
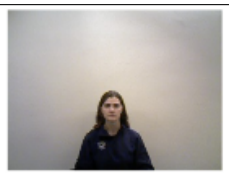
	Case 1: Naive Case, Face Mask	**Case 2: X-Translation, Sunglasses**
Value Image		
K-Means Output Cluster		
Segmented Occlusion Mask		
Final Reconstructed Image		

Table 4: Pipeline 1 Results for both Scenarios

3.2 Pipeline #2: Depth Image ICP Pipeline

Although the affine transformation provided good results, ICP was tested as another way of registering the non-occluded and occluded images.

An identical pipeline to Pipeline 1 (3.1) was implemented, with affine registration being replaced by ICP. Depth images were converted into 3D point clouds in order to compute ICP and then transformed back into a 2D depth image for the remainder of the pipeline. Results for x-translation and the naive case are shown below.

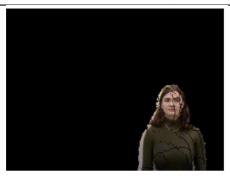
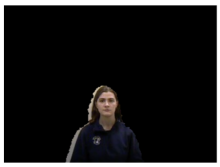
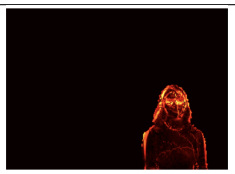
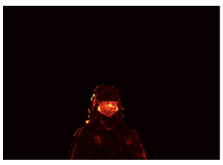
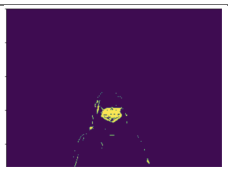
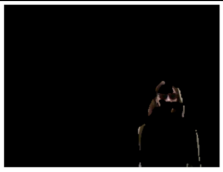
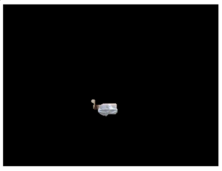
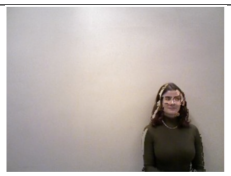
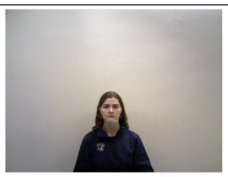
	Case 1: Naive Case, Face Mask	Case 2: X-Translation, Sunglasses
Translated 2D Depth Image		
Value Image		
K-Means Output Cluster		
Occlusion Mask		
Final		

Table 5: Pipeline 2 Results for both Scenarios

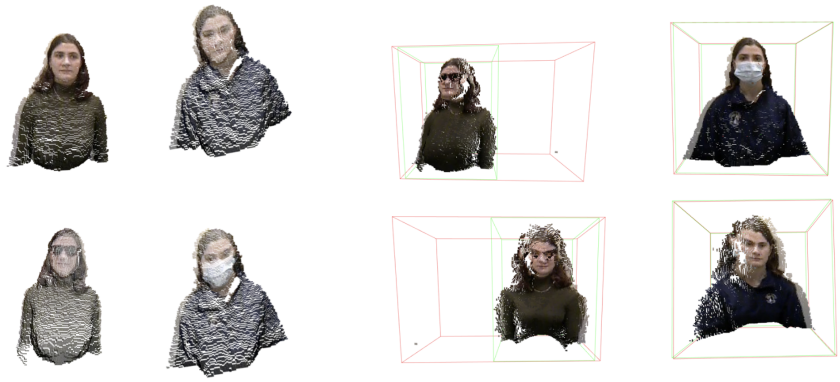


Figure 6: The left subfigure shows images as 3D point clouds: x-translation case (left) and no movement case (right). The right subfigure (b) displays registered non-occluded and occluded images: front view (left) and back view (right) of the aligned 3D point clouds, as shown in Open3D using ICP.

3.3 Pipeline #3: DBSCAN and Haar Cascade Pipeline

Pipeline 3 is identical to Pipeline 1 save for object segmentation. Using DBSCAN and face detection, the occluding object is identified.

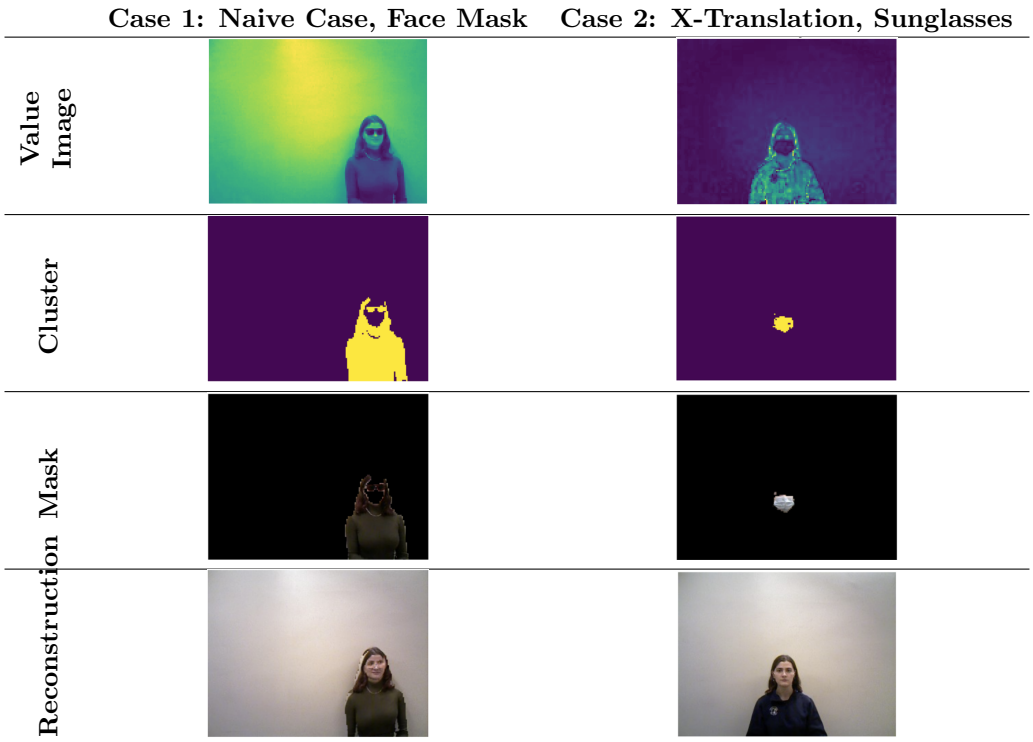


Table 6: Pipeline 3 Results for both Scenarios

3.4 Pipeline #4: Point-Cloud-only ICP Pipeline

After completing the first three pipelines, all of which utilized the depth images for reconstruction, there was interest in seeing the results if the entire pipeline was implemented using point clouds. This pipeline closely followed Pipeline #2 from 3.2 where the ICP algorithm was used

for registration and K-means algorithm was used for clustering. To maintain registration, the point clouds were voxelized to a size of 0.1 and the mask was derived using a KDTree for a nearest neighbours search to find regions where the RGB colours differed. The results of the x-translation and naïve case are below in Table 7.

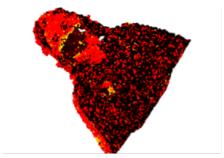
	Case 1: Naive Case, Face Mask	**Case 2: X-Translation, Sunglasses**
Value Image		
K-Means Output Cluster		
Segmented Occlusion Mask		
Final Recon-structed Image		

Table 7: Pipeline 4 Results for both Scenarios

4 Discussion

The four methods performed facial reconstruction to various degrees of success. At each step, there were multiple opportunities to and ways to approach the task. Ultimately, four pipelines were constructed by using different methods to solve the same problem. Applying the pipelines to sample input images made it possible to understand each one's strengths and deficiencies.

4.1 Pipeline #1: Naïve Pipeline

The Naïve Pipeline, which closely resembled the initial project plan and from which the subsequent pipelines were built, functioned well with simple input images but lacked flexibility and robustness. It required manual manipulation of images and could not automatically process

images from different data streams, even from the same acquisition. For example, the pipeline normalizes depth images, so background pixels have zero depth. While essential for registration, this normalization required the user to identify background depth values manually each time. As a result, the pipeline cannot process images with multiple subjects or non-plain backgrounds.

The ORB feature detection performed well, even under challenging conditions. However, the affine transformation, being rigid, struggled with small, non-linear movements. It required many, highly accurate ORB keypoints, which limited its effectiveness. When feature matching yielded a mean residual error over 25.0 pixels—such as during z-axis torso rotations—the transformation failed, rendering such scenarios unusable, as shown in 4.6.

When segmenting the occlusion mask, using the difference between the registered images as the value channel (brightness) to just get the occlusion, there were difficulties. Despite depth normalization, shadows during data acquisition caused background pixels to match the subject's depth, and RGB values couldn't distinguish them due to similar tones. This noise persisted even after K-means clustering, requiring thresholding to reduce it. 2.5 describes how morphological thresholding and median blurring were employed to extract the mask. These methods were effective but involved adjusting kernel sizes manually in four steps, limiting the pipeline's adaptability and automation.

While this pipeline works well for static scenarios or minor x-axis translations, its flexibility and robustness need improvement to handle diverse scenarios with greater accuracy.

4.2 Pipeline #2: Depth Image ICP Pipeline

This pipeline initially had promising results for both the naive case and the x-translation case. Fitness scores for the computed transformation between 3D point clouds was essentially perfect, and inlier RMSE values were less than 6, as shown in Table 10 in the Appendix. Visually, the point clouds looked well aligned when plotted, as shown in Table 5. Because this pipeline's registration utilizes all points during rigid registration, as opposed to aligning less than 50 matching key points found by ORB feature detection in the naive pipeline, the quantitative success of ICP's alignment for these simple cases makes sense. It should be noted that ICP was extremely sensitive to a manually chosen distance threshold and would likely have to be changed for more complicated transformation cases.

After registration was computed and point clouds were converted to 2D depth images, artifacts on the translated depth images were apparent in the x-translation case, as shown in the top row of Table 5. The cause of these missing values was not confidently determined but may have been caused by interpolation issues during projection. These artifacts lead to a worsened occlusion isolation from k-means, as the input to the algorithm (HSV brightness channel) was significantly affected by missing data and noise for the x-translation case. Although the segmented occlusion digital mask in both the naive pipeline and this pipeline appears to be similar, the final reconstruction was negatively impacted by the initial artifacts from the start of the pipeline. The final face in the x-translation case of this pipeline is much less recognizable.

4.3 Pipeline #3: DBSCAN and Haar Cascade Pipeline

DBSCAN and Haar Cascade provided accurate segmentation under the right conditions. Best results were seen on uniformly-coloured objects such as masks and gloves where objects could be detected irrespective of translation or rotation. In images with dark sunglasses, dark hair, and a dark shirt, DBSCAN struggled to segment the sunglasses and was extremely sensitive to small changes in hyperparameters. The resulting clusters are somewhat pixelated and while adhering to the shape of the occluding image, do not contain fine detail at the borders. Haar Cascade face detection proved accurate in most situations but occasionally failed in images where the mask covered most of the face below the eyes.

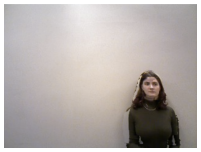
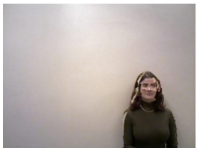
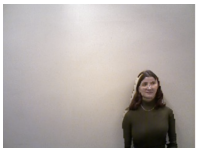
4.4 Pipeline #4: Point-Cloud-only ICP Pipeline

This pipeline struggled to provide accurate results. Even when voxelizing the point cloud, any error in the registration rendered this pipeline unusable because it was reliant on the RGB

colours being differentiable. After running the K-means algorithm, the occlusion could be mostly successfully segmented. The segmentation RGB value was then found and masked with the non-occluded region using a KTTree and Nearest Neighbours algorithm. Unfortunately, using nearest neighbours to accurately align the voxels distorted the fine facial features past recognition, even on the scenario with no movement. In the x-translation scenario, the sunglasses were at a different depth, so the nearest neighbours could not function for reconstruction. However, when the nearest neighbours was not used, the results were further corrupted – nearest neighbours rendered the best results. This pipeline is currently unusable and should be considered experimental.

4.5 Comparison of Pipeline Implementations

Table 8: Comparison of k-means clustering (top) and and final reconstruction images (bottom) for all pipelines

Naive	Depth Image ICP	DBSCAN	Point-Cloud ICP
			
			

K-means occlusion isolation and final reconstructed images for the x-translation case are shown in Table 8. K-means clustering of the sunglasses for the naive pipeline and the depth image ICP pipeline are relatively similar, both picking up the general shape of the glasses as well as unwanted regions like the hairline, arms and parts of the mouth. DBSCAN clustered the target's body along with the sunglasses, but in terms of the facial occlusion itself, it did the best job at computing a clean, closed segment of the desired region. K-means on the point-cloud in Pipeline #4 (3.4) performed the worst.

The best final reconstructed image was computed using the Naive affine pipeline (pipeline #1). The facial region itself had little to no noise or artifacts due to the registration or occlusion clustering; the subject's face was completely recognizable. Both the depth image ICP and DBSCAN pipeline's had semi-reconstructed images, but both had significant artifacts or misaligned features. The point-cloud ICP pipeline performed the worst with unusable results.

4.6 Other Scenarios

Table A in the Appendix section highlights two other test cases.

Z-axis Rotation

In the initial steps of our pipeline for the z-axis rotation case, there was already difficulty with computing matching features using ORB. Features between the original and rotated image were visually not corresponding, as shown in the top image of Table A. This caused the affine transformation to fail. Moreover, ICP failed to correctly align the two images, possibly due to the apparent deformation of the hair and arms in the rotated image.

Multi-Occlusion

In the case involving multiple occlusions, it should be noted that the k-means clustering using the HSV different image did isolate both the scarf and the sunglasses occlusions quite

successfully. However, there was no translation, rotation or zoom involved in this case. Despite this, registration using both methods did not perform that well, as seen by the displaced reconstructed nose.

5 Future Work

Currently, this implementation can only function on images with a single subject, taken against a plain background without other obstructions or objects. This implementation is also limited to small, unidirectional translations or rotations, as described in Section 4.6. In the future, more rigorous approaches such as non-rigid registration (opposed to the rigid registration methods in the current iteration) and voxel-based 3D keypoint matching for improved registration should be implemented such that the robustness of this algorithm is increased.

In addition, the autonomy of this algorithm will be enforced using current computer vision techniques. In the current implementation, some stages of the algorithm require manual interference, such as the ranges for depth normalization or which cluster from the K-means algorithm to use as the occlusion mask. Automating these steps is not improbable, the tools exist but were not used due to time limitations, but by implementing them and automating the pipeline for use on any appropriate image, the robustness and flexibility would be greatly improved.

Lastly, further data acquisition will be performed using an improved camera. The RGB-depth camera used, the PrimeSense Carmine 1.09, was manufactured in 2009 and is no longer in production – PrimeSense no longer updates this technology, making it obsolete. The default image resolution is 240x320 and no surviving documentation could be found to change the image resolution. Resultantly, many fine-grain details that are vital for facial reconstruction did not render correctly, making this problem difficult. In the future, a more robust RGB-depth camera with a higher image resolution will be used in data acquisition.

6 Conclusion

Using the PrimeSense RGB-depth camera, profile images were taking with and without occlusions. Four different pipelines were applied with various approaches to registration and segmentation and reconstructed facial images were constructed using the non-occluded pixels. The Naive affine pipeline was found to have the fewest artifacts and resulted in the most robust reconstruction. Lastly, ethical impacts must be considered. When facial reconstruction is applied to real world use cases, the technology can be used in situations where individuals might deliberately want to keep their face obscured. Access to a database of headshots could result in identification in scenarios constituting could be a breach of privacy. Overall, the initial results are promising, with further pipeline refinement needed to optimize performance in real-world applications.

References

[1] J. A. Mensah, E. , E. Ocran, S. Iddi, and L. Asiedu, "De-occlusion and recognition of frontal face images: a comparative study of multiple imputation methods," *Journal Of Big Data*, vol. 11, 04 2024.

[2] F. T. Zohra, W. Rahman, and M. Gavrilova, "Occlusion detection and localization from kinect depth images," *Conference: 2016 International Conference on Cyberworlds (CW)*, 09 2016.

[3] C. Low, A. Teoh, and C. Ng, "Multi-fold gabor, pca, and ica filter convolution descriptor for face recognition," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 29, no. 1, pp. 115–129, Jan. 2019, publisher Copyright: © 1991-2012 IEEE.

[4] W. Liu, Y. Wen, Z. Yu, M. Li, B. Raj, and L. Song, "Sphereface: Deep hypersphere embedding for face recognition," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 6738–6746.

[5] S. John and A. Danti, "Removal of occlusion in face images using pix2pix technique for face recognition," *Lecture notes on data engineering and communications technologies*, pp. 47–57, 01 2022.

[6] Y. Wang and J. M. Solomon, "Deep closest point: Learning representations for point cloud registration," *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2019. [Online]. Available: https://openaccess.thecvf.com/content_ICCV_2019/html/Wang_Deep_Closest_Point_Learning_Representations_for_Point_Cloud_Registration_ICCV_2019_paper.html

[7] Z. Daixian, "Sift algorithm analysis and optimization," in *2010 International Conference on Image Analysis and Signal Processing*, 2010, pp. 415–419.

[8] M. Muja and D. G. Lowe, "Fast approximate nearest neighbors with automatic algorithm configuration." *VISAPP (1)*, vol. 2, no. 331-340, p. 2, 2009.

[9] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: An efficient alternative to sift or surf," in *2011 International Conference on Computer Vision*, 2011, pp. 2564–2571.

[10] L. Li, L. Wu, and Y. Gao, "Improved image matching method based on orb," in *2016 17th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD)*, 2016, pp. 465–468.

[11] M. Werman and D. Weinshall, "Similarity and affine invariant distances between 2d point sets," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, no. 8, pp. 810–814, 1995.

[12] V. Ha and J. Moura, "Affine-permutation invariance of 2-d shapes," *IEEE Transactions on Image Processing*, vol. 14, no. 11, pp. 1687–1700, 2005.

[13] F. Wang and Z. Zhao, "A survey of iterative closest point algorithm," in *2017 Chinese Automation Congress (CAC)*. IEEE, 2017, pp. 4395–4399.

[14] J. A. Hartigan, M. A. Wong *et al.*, "A k-means clustering algorithm," *Applied statistics*, vol. 28, no. 1, pp. 100–108, 1979.

[15] R. Kumar, "A guide to the dbscan clustering algorithm," Datacamp.com, 09 2024. [Online]. Available: https://www.datacamp.com/tutorial/dbscan-clustering-algorithm

[16] OpenCV, "Opencv: Cascade classifier," docs.opencv.org. [Online]. Available: https://docs.opencv.org/3.4/db/d28/tutorial_cascade_classifier.html

7 Statement of Contribution

Work was contributed in equal portions by all group members.

Section	Person
Pipeline 1 Coding + Implementation	all
Pipeline 2 Coding + Implementation	Elise
Pipeline 3 Coding + Implementation	Sophie
Pipeline 4 Coding + Implementation	Isabel
GitHub Maintenance	Not Elise
Abstract	Elise
Introduction	Isabel
Motivation	Isabel
Current Models and Limitations	Sophie
What is our Solution	Isabel
Methods	Isabel
Data acquired and all the variations	Elise
Data processing	Isabel
Remove backgrounds	Isabel
HSV vs RGB	Isabel
Keypoint Detectors	Isabel
SIFT	Elise
ORB	Isabel
MSE	Sophie
Transformations	Sophie
Affine 2D	Isabel
ICP	Elise
Occlusion Segmentations	Sophie
Kmeans clustering	Elise
DBSCAN pipeline	Sophie
Reconstruction	Sophie
Results	Isabel
Pipeline 1 for Depth / Affine	Isabel
Pipeline 2 for DBScan	Sophie
Pipeline 3 for Point Cloud into Depth / ICP	Elise
Pipeline 4 for Point cloud / ICP, stays as PC	Isabel
Discussion	Sophie
Discuss Pipeline 1	Isabel
Discuss Pipeline 2	Sophie
Discuss Pipeline 3	Elise
Discuss Pipeline 4	Isabel
Compare Pipelines	Elise
Other Scenarios	Elise
Z-axis rotation	Elise
Scarf + Glasses (i.e.: multi occlusions)	Elise
Future Work	Isabel
Conclusion	Sophie

Table 9: Section and corresponding person responsible

A Appendix

Figure 7: Segmentation of a variety of occluding objects. Objects that performed the best had uniform colour, no specularity, and were distinct from their surroundings (see dark sunglasses hair up vs. hair down).

Table 10: Transformation matrix, fitness score and inlier RMSE of ICP registration for the translation case (left) and the no movement case (right).

Case	Fitness Score	Inlier RMSE
X-translation	0.99	5.30
No movement	1.0	3.43

Table 11: Other Cases; Z-axis rotation (left) and Multi-occlusion with scarf and sunglasses (right)