Localization of Botox Injection Points with AR

Natalie Lo[301198791], Aiden Carelse[301425491], Qiqi Liu[301256230], Edward Lai[301550876], and Kevin Zyfi[301389704]

{nclo, arc17, liuqiqil, cyl74, kzyfi}@sfu.ca
CMPT340 Fall 2024, Prof. Hamarneh

Abstract. This project report explores the application of Augmented Reality (AR) and Image Processing to enhance the localization of Botox injections to specific regions of the facial anatomy. Using MediaPipe's facial landmark detection feature, this project extracts real-time facial features and integrates these with predefined anatomical regions for accurate Botox injection guidance. Facial features such as eyes, noses, lips and facial contours are dynamically modelled using points and connections to represent the complete face structure. The database is used for tracking and managing data in detail which stores comprehensive information including patient profiles, treatment information, histories, injection key points, anatomical landmarks and segmentation data details. A user-friendly graphical interface developed with Streamlit, allows clinicians to visualize anatomical regions, injection points, and real-time overlays directly on facial images or video streams. Preliminary results demonstrate that the close integration of real-time AR visualization, structured databases, and an intuitive graphical interface provides an innovative tool for enhancing Botox injection accuracy, streamlining clinical workflows, and potentially improving clinician-patient communication.

Keywords: Augmented Reality · Facial Anatomy · Botox · MediaPipe · Streamlit · SQL

1 Introduction

Botulinum toxin (Botox) is widely used in medical and cosmetic procedures that require accurate positioning of the injection site to ensure efficacy and safety. However, individual anatomical variations and reliance on static schematics provide challenges for clinicians, increasing the risk of complications and undesirable outcomes. Recent advances in augmented reality (AR) provide opportunities to improve accuracy through real-time facial landmark detection and dynamic visualization.

This project integrates AR visualization, facial anatomy modelling, and database management into a unified system to improve injection accuracy, streamline workflow, and support clinicians in offering personalized treatment. The following section summarizes the problem description, motivation, background, and related work, as well as a report structure map.

1.1 Application and Problem Description

Botox injections not only smooth or soften dynamic facial wrinkles, but they also shape and enhance facial contours [1]. However, achieving precise and safe Botox injections requires an in-depth understanding of facial anatomy and superior technical skills. Incorrect injections can lead to complications such as bruising, headaches, muscle twitching, drooping eyebrows or eyelids, or over-elevation of the eyebrows [2], which can impact negatively patient outcomes and satisfaction. Traditionally, clinicians have relied on static anatomical images or their previous knowledge to identify injection sites. Although this method works well for experienced practitioners, it is limited by the fact that facial anatomy varies greatly from person to person. Emerging technologies, such as AR, offer new opportunities to improve the accuracy and personalization of Botox injection procedures.

1.2 Motivation and Background

AR technologies offer a hopeful solution for improving Botox injection accuracy by enabling real-time visualization of anatomical landmarks on a patient's face. The inclusion of a user-friendly graphical user interface (GUI) enhances procedural confidence by allowing clinicians to visualize anatomical areas and injection sites dynamically. Meanwhile, a structured database ensures efficient management of patient data, injection history, and anatomical landmarks. The combination of AR, GUI, and a robust database not only improves injection accuracy but also improves clinical workflow and easy retrieval of patient records and treatment history. This integration satisfies the growing demand for precision medicine and data-responsive healthcare solutions.

1.3 Related work and Literature Survey

The use of AR in the medical field has grown significantly in recent years. Several studies have explored techniques related to injection site localization. Chen and others [3] showed the application of AR for acupoint localization, which improved the accuracy and usability of AR for medical practitioners. Rhee and others [4] reviewed advances in the use of machine learning for facial marker detection, highlighting its role in enhancing real-time facial feature analysis. Additionally, Small [5] emphasized the importance of accurate Botox injection localization and the challenges posed by anatomical variability. Despite these progresses, there is still limited research dedicated to the integration of AR and structured data management for Botox injection procedures. This project aims to use these techniques by integrating them to enhance Botox injection procedures.

1.4 Report Structure Map

The rest of the report is divided as follows: Section 2 describes the materials used, such as facial images and data management strategies. Section 3 details

the methods employed, covering facial landmark detection, anatomical mapping, AR visualization and GUI implementation, database integration and data flow and system architecture. Section 4 presents the results, while Section 5 outlines the accomplishments of the project. Section 6 discusses individual contributions to the project. Section 7 presents the conclusions and discussions, providing insights into the strengths and limitations. Finally, Section 8 outlines future work, suggesting potential improvements and extensions.

2 Materials

This section provides an overview of facial images, segmentation and injection data used in the project.

2.1 Facial Images and Segmentation

The primary data used in this project consists of facial images. Facial images are collected from live webcam streams and pre-uploaded images. Google's API Mediapipe Face Landmark Detection library detects face expression and landmarks in real-time and images. These landmarks are plotted against distinct facial features such as eyes, nose, lips and contours represented by key points. Mediapipe's Face Detection Task served as the base of mapping appropriate regions and injection points in the program by utilizing their normalized coordinates. The library does not provide plotting polygons on the face mesh, therefore segmenting the regions were performed manually using CVAT. (Computer Vision Annotation Tool). using its image annotator tool. The data was then exported and processed in Python to integrate them into the program and to the database(See Fig. 1). The segmented data is stored in a structured format in a database to ensure consistency of facial region and injection point references.

2.2 Injection Data

The data used in this project was obtained from several key clinical references. Typical examples of injection data include injection name which is based on injection regions such as points on the frontal muscle[6], type of treatment[5], dosage[7], depth[8], the desired outcome such as softening of forehead horizontal line[6], and side effects such as eyebrow ptosis[6]. This data is organized into a structured format to guide injection planning and improve safety and efficiency. Clinical insights from the literature were integrated into an AR-based system to ensure safety and efficiency.

2.3 Summary

These elements work together to ensure the accuracy of the injection. The following section discusses the methods used to construct a comprehensive system for visualizing Botox injection in detail.

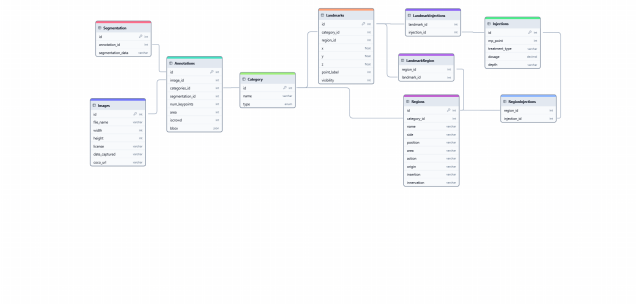

Fig. 1. Database schema for facial landmark detection and anatomical mapping.

3 Methods

This section details the methods used for facial landmark detection, anatomical mapping, and AR visualization, and how these components are integrated into the Botox injection guidance system. It includes the implementation details of each module, the algorithms and tools used, and the overall system workflow.

3.1 Facial Landmark Detection

Facial landmark detection forms the foundation of this project. Mediapipe's Face Landmarker model was used to detect key facial landmarks from both static images and live video streams. The model produces 478 unique landmark points on the face, including important regions such as the eyes, lips, nose, and jawline. ARBotox's implementation uses the landmarks detected from the face landmarker in order to plot and update the segmentation of the facial anatomy in video and image, overlaying relevant anatomical information. Mediapipe's framework [9] provides a comprehensive environment for building perception pipelines, enabling efficient and reproducible results across different devices and platforms.

In our development phase, Landmarker.py was used to visualize and customize the output of Mediapipe's face mesh such as plotting and labeling landmark values by range and region, and displaying text annotations. The main classes and functions used in the 'Landmarker' module are shown in

3.2 Anatomical Mapping and AR Visualization

The next step is mapping the detected landmarks to predefined anatomical regions. To associate a value from a detected landmark to a facial anatomical

region, manual mapping was performed to produce segmentation data representing boundaries of the facial anatomy. Each detected landmark is associated with a corresponding region, enabling precise localization of injection points.

3.3 Image Annotation

The first step in image annotation involved using Mediapipe's face mesh on an image of a front-facing face with neutral expression and then exported as a .png. The image was then imported into CVAT to perform manual annotation of both polygon and keypoint data by annotating directly on the image with the face mesh to trace the relative landmark locations lined up with Mediapipe's face mesh. CVAT provides an attribute labeler for each annotation to which was used to map a label of anatomical regions to the coordinate values of polygons and keypoints.

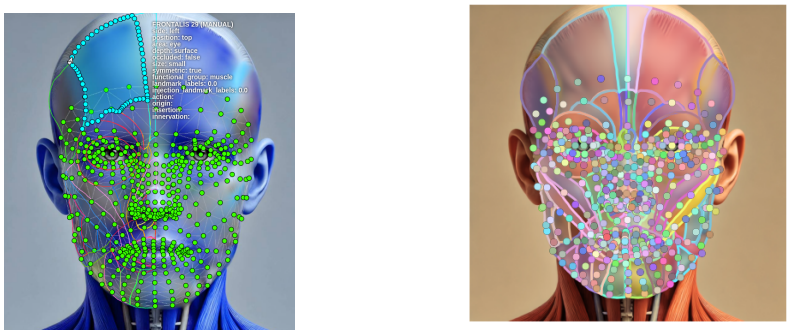

Fig. 2. (left) Image annotation in CVAT. The green dots are manual keypoint annotations which traced Mediapipe's point landmarks. (right) Annotation results plotted in a notebook after normalizing coordinate data

3.4 Data Extraction and Preprocessing

Following annotation, the annotations were exported as .json and .xml files containing annotation data captioned in a COCO dataset (Common Objects in Context). This process was performed on the notebook file `Data_Extraction.ipynb` The notebook details the process of importing the raw files and extracting the annotated data to Pandas dataframe, normalizing and adjusting coordinates, plotting and visualizing the resulting data, and exporting the data into a `.csv` file.

3.5 Data Transformation and Feature Engineering

The data was imported into another notebook, `Feature_Engineering.ipynb`, for feature engineering. This step involved cleaning and organizing feature labels

in a DataFrame. Unnecessary columns generated during the annotation step and export from CVAT were removed. New DataFrames were created and manipulated to extract data relevant to ARBotox. The notebook outputted a new set of `.csv` files containing the engineered features and values of facial regions and their normalized polygon coordinates, normalized keypoint values which are mapped to Mediapipe landmarks, and mappings of groups of keypoints which fall into specific polygon segments. The mapped data was then visualized by plotting the results in the notebook, showing keypoints and segmentation labeled with the facial anatomy. These files were then imported into the program to be used for projecting segmentation and nodes onto the face mesh.

3.6 Program Design and Implementation

The main logic of the program is primarily implemented by `ArBotox.py` while the file `DataDriver.py` manages the saving and loading of files between the program and database. The following table provides a brief description on the classes and functions role in implementation. The `AnatomyRepository.py` provides an abstraction to encapsulate direct data access between the program and the database.

Type	Name	Description
class	DataDriver	Responsible for communication of data-related tasks between the database and the main program, such as loading files, and saving and updating dataframes into the database through communication with AnatomyRepository.
class	AnatomyRepository	Responsible for direct access to the database. Defines classes, serializers, getters, and setters to create, retrieve, update, and delete anatomy-related records.
class	ArBotox	The main class, which holds all functionality, such as computing appropriate coordinate values to plot, drawing functions, and organizing data received for output.
class	DrawOptions	This class handles options input to the program for which segments and labels to plot. It helps sorting and organizing what options were passed
function	apply_draw_options()	Function filters selected options
function	landmarks_to_dataframe()	Converts the detected mediapipe landmarks into a dataframe
function	detect_and_draw()	Detects the landmarks and performs all drawing.
function	draw_points()	Draws the landmark points and labels.
function	detect_on_stream_proto()	Driver for video feature.
function	detect_on_image_proto()	Driver for image feature.

Table 1. ArBotox Classes and Functions

The output of ARBotox effectively projects the selected anatomical facial regions with their labels as well as selected treatment types. The white mark-

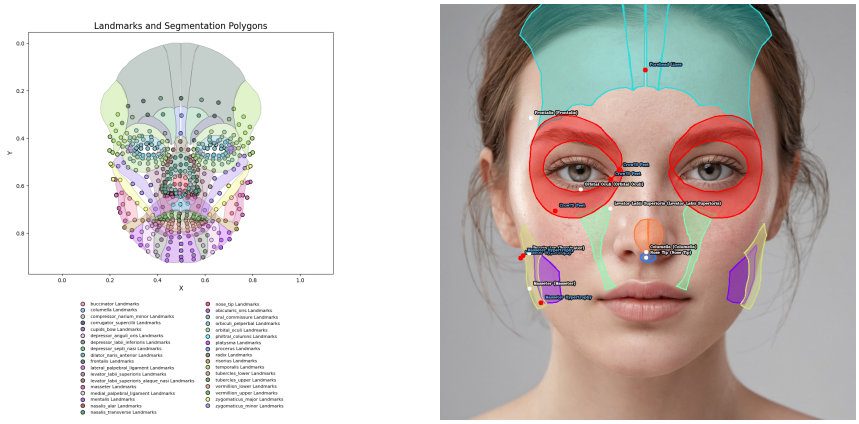

Fig. 3. (left) Image annotation in CVAT. The green dots are manual keypoint annotations which traced Mediapipe's point landmarks. (right) Demonstration of ARBotox and labeling of facial regions and treatment injection point

ers represent point-labeling for facial regions while the red markers represent injection points, labeled with the injection name as well as the facial region.

3.7 GUI Design and Implementation

The GUI is a key component that integrates all functionalities of the system, providing a seamless experience for clinicians. The GUI is developed using Streamlit, which enables easy deployment and interaction. The interface allows clinicians to upload images, view real-time video streams, visualize facial landmarks and injection points, and manage patient data.

The GUI consists of several tabs, each serving different functions(See in Fig. 4): Real-Time Feed: Shows the AR-enhanced video stream with overlaid anatomical landmarks and injection points, providing real-time guidance during the injection procedure. Patient Information: Allows clinicians to add or search for patients, view detailed profiles, and update treatment history. Injection Points: Displays the segmented regions of the face and allows clinicians to select specific areas for injections.

The GUI is designed with usability in mind, ensuring that all functions are easily accessible and that clinicians have a clear view of both patient information and AR visualization. This integration of various components through the GUI significantly enhances workflow and ensures accurate and efficient Botox treatment.

3.8 Database Integration

The system uses a relational database to manage patient data, injection records and anatomical landmarks. The database is implemented in MySQL, with tables

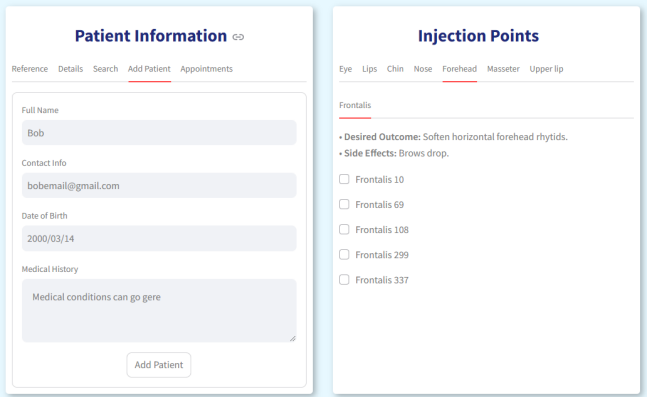

Fig. 4. GUI Display for Patient Information and Injection Points

for managing patients, appointments, treatment history, and injection points. Database integration is handled through SQLAlchemy, which serves as an Object Relational Mapper (ORM) to facilitate interaction between Python code and the MySQL database [10]. Fig. 5 illustrates the database structure that supports the entire facial landmark detection and anatomical data management system, ensuring robust and efficient data storage. In addition, the DataDriver class is an important part of the system's data processing capabilities and it is the main tool for managing the flow of data between CSV files database and other components. It works together with Anatomy Repository to perform various operations, such as saving nodes, regions and injections to the database, as well as establishing relationships between different entities. The class also plays a key role in maintaining data consistency by converting data between Pandas data frames and SQL database formats. These data frames help in subsequent processing and analysis. Real-time data entry from the GUI, coupled with a structured database and efficient data management tools such as Panda and SQLAlchemy, ensures that the system remains accurate and current in all phases of interaction, from adding patients and managing appointments to documenting treatment history.

3.9 Data Flow and System Architecture

The system architecture involves several interconnected components: data acquisition, image processing, AR visualization, database management, and the user interface. This architecture ensures a streamlined workflow, integrating all components to provide a comprehensive AR-based tool for Botox injection guidance.

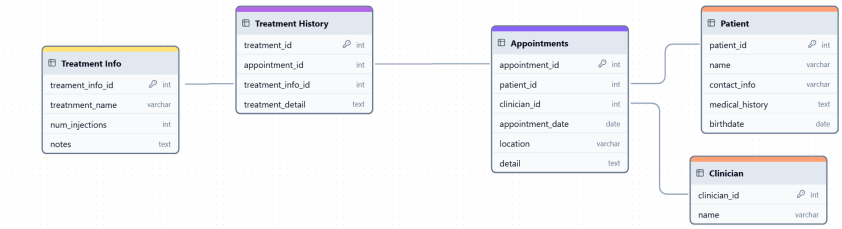

Fig. 5. Database

4 Results

The AR-based Botox injection system successfully integrates facial landmark detection, segmentation and injection point visualization. The Fig. 6 below shows the system in action, highlighting key facial features detected by the MediaPipe Face Mesh model, real-time anatomical mapping, and the injection guidance overlay. Screenshots(See Fig. 7) of the graphical user interface (GUI) demonstrate the clinician's workflow, including patient data management and injection point selection. These visuals demonstrate the system's real-time performance and user interactivity, enhancing clinical decision-making during Botox procedures.

5 Accomplishments

During the entire project development process, major advances were made in the integration of AR technology with real-world clinical applications. MediaPipe's facial landmark detection combined with anatomical mapping and AR visualization proved to be an effective way to improve the accuracy of Botox injections. The development of a functional GUI allowed users to actually interact with the system, thus enabling clinicians to use the tool easily and efficiently. Despite the challenges in exactly partitioning facial regions and the complexity of database integration, these obstacles were overcome to a large extent by moving to a more

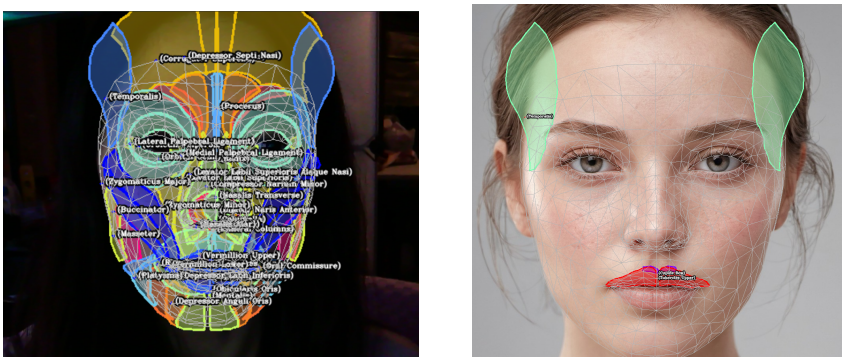

Fig. 6. Display of Injection Region and Injection Point

structured annotation approach using CVAT and utilizing SQLAlchemy for our database interactions.

However, not all challenges have been fully addressed. Achieving perfect injection point localization for all face shapes is still difficult due to anatomical differences. Further improvements to the model as well as expansion of the dataset could improve this in future iterations. This project nevertheless resulted in an effective AR guidance tool that can effectively overlay injection points and has a reliable database and user interface to support clinical use.

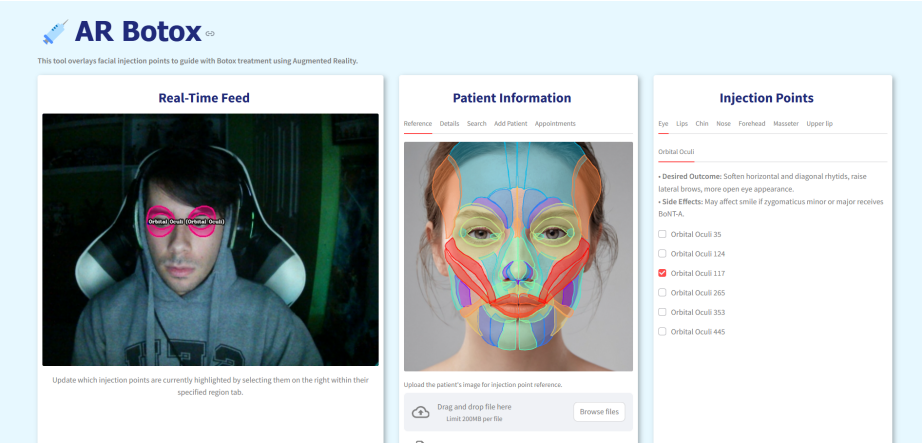

Fig. 7. Complete GUI Display with 2D and 3D Mapping

6 Contributions

Natalie Lo: Researched information on Botox, algorithms and computer vision related tasks in order to implement our project. Pre-processed data required for the plotting on polygons and point values on the webcam such as image annotation, mapping of coordinate values of polygons to Mediapipe's landmarks, gathering and normalizing coordinate values, feature engineering, and importing all pre-processed data to integrate with the program. Wrote the program's logic in python to effectively display and update polygons, landmarks, and labels in images and webcam. Wrote the necessary python files to integrate the local database and the main implementation to the program.

Aiden Carelse: Designed and implemented the basis for the front-end. Worked on connecting the back-end provided by Natalie to the GUI by displaying the real-time feed on the GUI. Set up functionality to import 2D reference images from the GUI, once again using the functions Natalie wrote. Added the patient details and search tab, which Kevin/Edward expanded upon when connecting with the database. Implemented the injection points tab, allowing user selection, and organizing all of the landmarks and displays pertinent information provided by Natalie and Qiqi. Completed the README file, helped with the report, and aided Edward with the demo.

Qiqi Liu: Based on the tables posted by Natalie, updated the tables in the database and then handed them over to Edward for finalization. Researched Botox injection data, and organized and completed injection data. Completed most of the report.

Edward Lai: Created the local MySQL database for the GUI and assisted Qiqi in database design for Natalie, as well as providing research and diagram for the injection points. Created mock data for testing purposes. Created demo video.

Kevin Zyfi: Connected the MySQL database with the GUI using SQLAlchemy, building on the work done by Aiden and Edward. Added functionality to the GUI elements for displaying patient info and searching for patients. Created tabs to add new patients and new appointments. In the appointments tab, there is a way to add specific treatments for each appointment. History of a patients appointments (past and future) is also shown. Created simple functions to gather injection, landmark, region, and image data from the database. Helped polish the README.

7 Conclusion and Discussions

In this project, we successfully developed a GUI using Streamlit by combining facial landmark detection, anatomical mapping and AR visualization. This allowed clinicians to visualize injection points, manage patient information and interact in real-time. The structured MySQL database integration guaranteed continuous access and easy updating of patient records, treatment histories, and landmark data.

Despite these achievements, the current project has several limitations that must be addressed. The project is limited in that it relies on pre-trained models that may not be personalized enough to accommodate the diversity of facial structures in clinical practice. In addition, real-time performance could be optimized to reduce latency further and improve the usability of live patient conversations.

8 Future Work

Future work on this project may relate to multiple improvements and ways to extend it:

3D mapping and visualization: Extending the system to support 3D facial landmark detection and visualization will improve the accuracy of AR overlays. By utilizing 3D data, the system could provide more realistic and spatially accurate guidance for Botox injections.

Interactive Feedback System: Implementing features that allow clinicians to give real-time feedback will make the system more interactive. For example, the system could allow clinicians to dynamically annotate or adjust injection points and feed them back into the model to improve future predictions.

Integration with other sensors: Integrating external sensors, such as heart rate monitors or skin temperature sensors, can provide additional data points for personalized treatment plans. Combining AR visualization with real-time physiological monitoring can provide a more complete approach to patient care.

These recommendations, as well as continued improvements to the model and user testing, could help push the way for the development of more comprehensive and interactive AR-assisted healthcare tools that can support a range of clinical applications beyond botulinum toxin treatment.

Acknowledgements

We are sincerely thankful to Professor Ghassan Hamaneh for his guidance and support throughout this project. His insights have been invaluable in pointing us in the direction of our research. We would also like to thank our team member Natalie Lo for her work with MediaPipe which laid the foundation for our understanding of the potential of image-based interactive communication in healthcare settings. inspired us to explore the capabilities of MediaPipe in this application. Finally, we would like to thank all of our team members for their discussions, suggestions, and encouragement, providing valuable feedback during the development of the project.

References

1. L. Farolch-Prats and C. Nome-Chamorro. Facial contouring by using dermal fillers and botulinum toxin a: A practical approach. *Aesthetic Plastic Surgery*, 43:793–802, 2019.

2. N. J. Lowe et al. Dosing, efficacy and safety plus the use of computerized photography for botulinum toxins type a for upper facial lines. *Journal of Cosmetic & Laser Therapy*, 12(2):106–111, 2010.
3. Y.-Z. Chen et al. Localization of acupoints using augmented reality. In *Proceedings of the 8th ACM on Multimedia Systems Conference (MMSys'17)*, pp. 239–241, 2017.
4. C. Rhee et al. Augmented reality can improve accuracy in identifying botulinum toxin injection sites. *EMJ Innovations*, pp. 25–32, 01 2022.
5. R. Small. Botulinum toxin injection for facial wrinkles. *American Family Physician*, 90(3):168–175, Aug 2014.
6. J. B. Kaplan. Consideration of muscle depth for botulinum toxin injections: A three-dimensional approach. *Plastic Surgical Nursing: Official Journal of the American Society of Plastic and Reconstructive Surgical Nurses*, 37(1):32–38, 2017.
7. J. Carruthers et al. Consensus recommendations on the use of botulinum toxin type a in facial aesthetics. *Plastic and Reconstructive Surgery*, 114(6):1S–22S, 2004.
8. Medscape. onabotulinumtoxina (rx): Brand and other names: Botox, botox cosmetic, botulinum toxin, 2024. Accessed on November 28, 2024.
9. C. Lugaresi et al. Mediapipe: A framework for building perception pipelines. *arXiv preprint arXiv:1906.08172*, 2019.
10. M. Bayer. Sqlalchemy. In A. Brown and G. Wilson, editors, *The Architecture of Open Source Applications Volume II: Structure, Scale, and a Few More Fearless Hacks*. aosabook.org, 2012.