Generation of Annotated Data for Human-centric Computer Vision Tasks Using Augmented Reality

Alexandros K. Vasileiadis, Nikos Nikolaidis
School of Informatics
Aristotle University of Thessaloniki
Thessaloniki, Greece
{vkalexan,nnik}@csd.auth.gr

***Abstract*—An Augmented Reality (AR) application for generating datasets for human-centric computer vision tasks such as human body and head detection or face recognition is presented in this paper. The application allows users to place virtual, static or animated, humanoids in real-world environments and capture, in an automated way, mixed (real + synthetic) images under various conditions, including different distances from the camera, and viewpoints. It can then export both the images and the corresponding metadata, including ground truth bounding boxes. Both the application and the generated dataset, which was created by placing each of 19 models in 11 different physical locations are publicly available and can be used to train and evaluate computer vision algorithms, including active vision ones. Face recognition experiments performed upon these data reveal expected performance variations under different (bright/dark) lighting conditions, distances (near/far), and view angles (0°–360°).**

***Index Terms*—Augmented Reality, Face Recognition, Dataset Generation, Synthetic Data, Computer Vision**

I. Introduction

Human-centric computer vision tasks such as human detection and recognition find numerous applications in areas such as surveillance, robotics, and human-computer interaction. Deep learning algorithms [1]–[3], have achieved remarkable success in these tasks. However, the performance of these algorithms heavily relies on the availability of large and diverse datasets for training and evaluation.

Traditional dataset creation methods for such algorithms often involve capturing images or videos of human subjects in real-world. This process can be time-consuming and resource-intensive, especially when annotations are to be included. Furthermore, they may raise privacy concerns. An alternative that effectively addresses these challenges is the use of Augmented Reality (AR). AR [4] enables the overlay of properly lit virtual objects onto the real world, creating a mixed reality environment. Indeed, by leveraging AR one can systematically place virtual humanoids in real-world environments, enabling dataset generation without involving human subjects. This approach offers several advantages, including:

Efficiency: AR allows for rapid dataset generation, as virtual humanoids can be easily placed and manipulated in the AR environment.
Accessibility: AR technology is becoming increasingly accessible through smartphones and tablets, making dataset creation possible for a wider range of users.
Control: AR provides precise control over the generated data characteristics, including humanoid location and orientation in the environment, enabling the creation of datasets tailored to specific research needs.
Automatic metadata creation: Metadata such as subject ID, location and orientation with respect to the camera, as well as bounding box, are readily available.

In this paper, we present an AR application developed in the Unity game engine for generating mixed (real + synthetic) datasets for computer vision tasks such as human body and head detection, face recognition or activity recognition. The application allows users to place virtual humanoids in real-world environments and capture images by varying camera-subject distance and view angle in a systematic way. Moreover, it exports both the images and the corresponding metadata, including ground truth bounding boxes for the humanoids. The generated data are particularly suitable for training and testing active[1] and view-independent human-centric vision approaches.

To evaluate the application's effectiveness, we used it to create a dataset by placing each one of 19 realistic humanoid models in 11 different physical locations and conducted face recognition experiments, analyzing how factors such as lighting, distance, and rotation impact recognition accuracy. The application as well as the generated dataset are publicly available.

The remainder of the paper is structured as follows: Related work is presented in Section II. Section III provides an overview of the AR application and its functionalities. Section IV details the technical implementation, including the automated procedure that can be used to generate datasets and the metadata that are exported. Section V describes the generated dataset characteristics, while Section VI presents the face recognition experiments and analyzes the results. Finally, Section VII concludes the paper.

[1]Active perception exploits the ability of e.g. robots to interact with their environment, for example move in space, towards increasing the quantity or quality of information obtained through their sensors and improving their performance in various perception tasks, such as human body detection of face detection and recognition

II. Related Work

Synthetic data, or combinations of such data with real-world data, are nowadays frequently used to train computer vision algorithms. The work in [5] presents an AR-based approach for dataset generation, focusing on enhancing object detection models by augmenting real-world images with synthetic objects. A method that can generate annotated video data depicting flying UAVs, using as input real background videos and 3D UAV models is described in [6]. The data can be use for training or testing UAV detection algorithms. In [7] the authors present a method that generates realistic mixed data with annotations for person detection, face recognition and human pose estimation. The method takes as input real background images and populates them with 3D human models in various poses. Both this and the previous paper use an approach similar to that used in AR: they use a virtual camera to capture 3D models properly scaled and placed in front of real images or videos in a 3D environment. However, the methods mentioned above don't allow systematic control over parameters such as subject placement, distance, and rotation within the scene, nor do they provide applications[2] for structured dataset generation. In contrast, our approach includes a dedicated, easy to use Android application that enables precise manipulation of humanoid models' positions and orientations in real-world environments, captured by the user. This allows for systematic dataset generation and controlled experimentation.

III. Application Description

The developed AR application provides an interactive and flexible approach to mixed data generation, by allowing users to place, in a controlled manner, realistic synthetic human models in real-world settings. Implemented in Unity, the application enables the placement, manipulation, and capture of virtual humanoids in different conditions to facilitate research in human detection, face recognition and related tasks.

The application operates in two modes: real-time capture and playback mode. In real-time mode, users can place virtual models in their physical environment, adjusting factors such as view angle and camera-model distance before capturing images. In playback mode, user-recorded AR sessions (videos captured through the application, that also carry related metadata) can be replayed to generate in an automated way datasets under controlled conditions.

Users can perform a number of operations and manipulate various parameters such as:

- Model selection: Different humanoid models can be chosen to ensure dataset diversity.
- Pose and animation: Models can be static or animated, to simulate walking and running.
- Viewing angles: The application allows systematic variation of humanoid orientation.A full 360° circle is performed for each model, using user-defined increments.

[2] [7] provides code that can be manipulated to generate data but the procedure is not straight-forward.

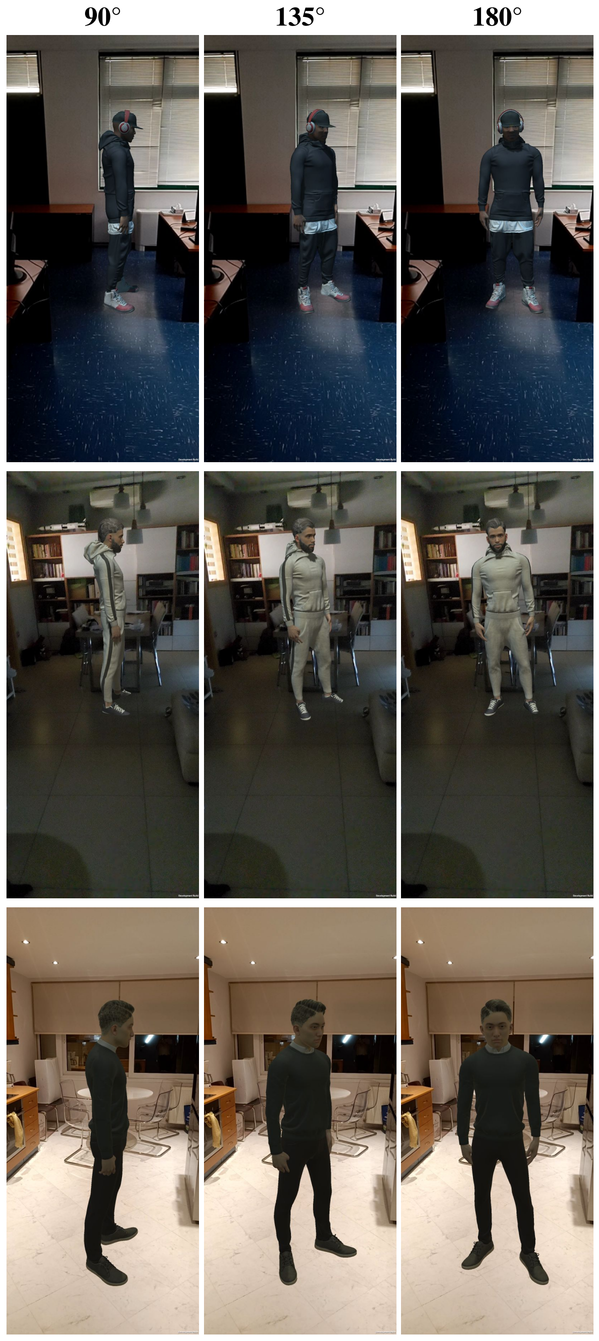


Fig. 1: Examples of humanoid models captured under different conditions (lighting, distance, view angle) and orientations in various physical environments.

- Distance from camera: The user can adjust the placement of models at different distances from the camera to produce diverse data or analyze the impact of distance on recognition accuracy.

The application automatically captures images and exports metadata, including camera parameters as well as bounding box annotations for the head and the body. This structured approach ensures high-quality datasets suitable for machine learning model training and evaluation. By enabling automated data collection and annotation, this tool significantly reduces the manual effort required in dataset creation while providing high levels of realism, control and reproducibility.

IV. Technical Implementation Details

The AR application was developed for Android devices, using the Unity game engine [8], leveraging AR Foundation [9] and ARCore [10] to provide a scalable and flexible framework for dataset generation. The following subsections detail the key components of the implementation, including 3D humanoid models, data annotation, and automated dataset generation.

A. Development Framework

Unity was chosen for its cross-platform capabilities and extensive support for AR development. The AR Foundation framework serves as an abstraction layer, enabling compatibility with ARCore (Android) and ARKit (iOS). The core functionalities used in this project include [11]:

- **Plane Detection:** Used to identify flat horizontal surfaces, in our case the ground, where humanoid models are placed.
- **Raycasting:** Employed for accurate model positioning and interaction.
- **Light Estimation:** The estimated natural light parameters were used to adjust humanoid model lighting so as to match real-world illumination, improving realism.
- **AR Recording and Playback:** This functionality allows the capture and subsequent replay of videos of real-world environments, along with the required metadata, thus facilitating off-line dataset generation at any time.

B. 3D Humanoid Models and Animations

The application utilizes 19 3D royalty free realistic humanoid models with diverse characteristics, ensuring dataset variability. The models along with the associated animations, were obtained in FBX format from Adobe's Mixamo [12] and were selected based on the following criteria:

- Different **body structures** and clothing to mimic real-world diversity.
- Pre-rigged **animations** (idle, walking, running) to simulate natural human movement.

Each humanoid model is assigned a unique identifier, that is stored in the metadata file alongside its spatial and appearance attributes.

C. Automated Dataset Generation

To streamline dataset creation, the application automates the process of model placement, movement, and image capture. The corresponding procedure consists of the following steps:

1) A humanoid model is spawned on a plane detected in the environment, at a certain distance from the camera. This distance is evaluated automatically so that the entire model fits (in terms of height) in the camera view. Alternatively, the model can be placed at a location selected by the user through touch and raycasting.
2) The model undergoes systematic rotation using user-defined increments (e.g., steps of 45°) so as to capture it from a series of viewpoints.
3) After completing a full 360° rotation, the model moves away from the camera by a user-defined distance increment (e.g. 0.5m) and the rotation step is repeated.
4) The procedure (model 360° rotation followed by translation) is repeated a number of times (e.g. 3), as defined by the user.
5) Screenshots are captured at each step, while metadata (bounding boxes, lighting conditions, distance) are recorded.

This automated approach ensures consistency and reproducibility in dataset generation.

D. Bounding Box Annotation and Metadata Export

Each captured image is annotated automatically with precise ground truth bounding box information (using Unity's built-in coordinate system) as well as other information. The metadata are stored in a structured CSV file and include the following fields:

- **3D Head and Body Bounding Boxes:** Defined in world space.
- **2D Head and Body Bounding Boxes:** Computed by projecting the 8 vertices of the 3D bounding box onto the camera's screen space and determining their enclosing axis-aligned rectangle by selecting the minimum and maximum coordinates for the top-left and bottom-right corners.
- **Model ID:** A unique identifier for each humanoid model used in dataset generation.
- **Model Rotation:** The degree of rotation applied to the humanoid model at the time of capture.
- **Distance from Camera:** The measured distance (in meters) between the camera and the humanoid model.
- **Screenshot Filename:** The filename of the corresponding image. If animation capture is disabled, the filename includes the model ID, its distance from the camera and the rotation degrees e.g., *0_3_0.jpg* for Model ID 0, a camera-model distance of 3 meters and a rotation of 0^0 . If animation capture is enabled, the format also includes the animation type and frame number, e.g., *0_3_0_StandardWalk_frame1.jpg* for a walking animation at frame 1.

It is important to note here that the procedure that was used to generated the 2D bounding boxes might cause them to be somewhat larger than the true 2D shape. This effect is more prominent in views from certain angles, such as 45^0 and 135^0.

These metadata elements ensure that researchers can accurately map experimental conditions to recognition performance results. Bounding boxes serve as ground truth annotations for object detection and recognition models, while rotation and distance parameters facilitate the evaluation of algorithm robustness, or their training, under different viewing conditions.

By integrating AR capabilities with an automated annotation pipeline, the application provides a scalable and reproducible solution for generating mixed datasets tailored for human-centered computer vision research.

V. Dataset Description

The dataset generated by the AR application consists of a large number of images and corresponding metadata. The dataset includes the 19 humanoid models, each positioned in turn in 11 different real-world environments (10 indoor, such as rooms or offices and 1 outdoor), in two different distances, 8 different view angles (0^0 to 360^0 in 45^0 increments) and under two different lighting conditions (bright and dark, obtained by capturing the videos of the various physical locations in two different times of the day or by using the lights). Thus, the dataset allows researchers to evaluate how different factors affect the accuracy of human body, head, and face recognition algorithms. Sample images are presented in Fig. 1.

The dataset is structured in a hierarchical format, with separate directories for different experimental conditions. Metadata files are stored alongside images, ensuring easy accessibility and traceability of dataset components. The AR application as well as the dataset are publicly available[3] to encourage further research and benchmarking in the field of computer vision.

VI. Face Recognition Experiments

To evaluate the usefulness of the proposed application and the generated dataset, we conducted face recognition experiments using a deep learning-based face recognition model. The primary objective was to assess how different viewing conditions, such as lighting, distance from camera, and head orientation, impact recognition accuracy.

A. Experimental Setup

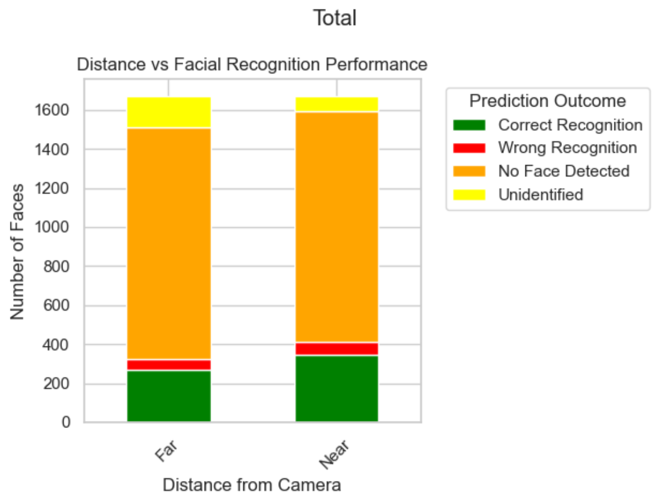
We utilized the face recognition library in [13], which is based on dlib. The model was pre-trained on real-world data and there was no fine-tuning on data from our dataset. The model was given access to a gallery containing one frontal close-up face image for each of the 19 humanoids present in the dataset. The evaluation included the scenarios present in the dataset. Humanoids face images were presented to the model in dark and bright lighting conditions, small (near) and larger (far) distances from the camera (the two locations differ by 1 meter), and 8 different viewing angles (0^0-360^0 in 45^0 increments, the frontal view being that of 180^0).

The model's results were categorized into four groups and the respective percentages were calculated:

- Correct Recognition: The model correctly identified the face and matched it to a known identity in the galery.
- Wrong Recognition (Misidentification): The model identifies a face but incorrectly assigns it to a wrong identity in the galery.
- No Face Detected: The model fails entirely to detect a face.
- Unidentified: The model detects a face but cannot match it to any identity in the gallery.

Results are presented in Figures 2 and 3, in the form of barcharts.

[3]https://github.com/Alexandros-Vas/AR_Humanoid_App



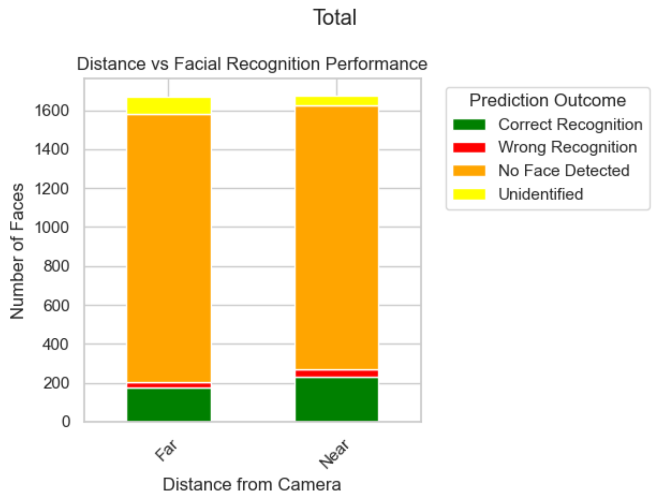
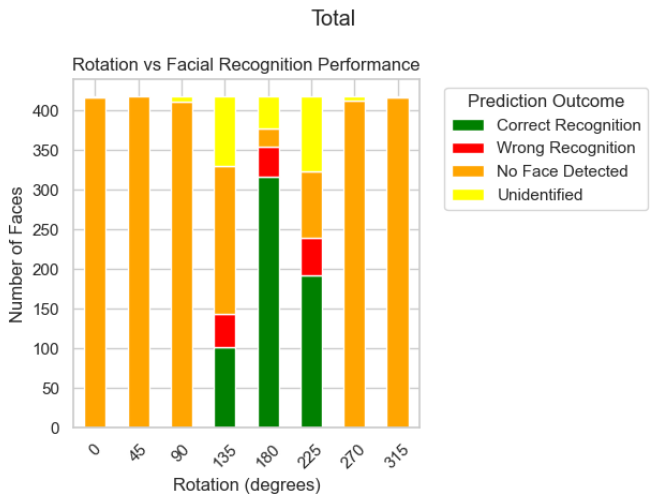


Fig. 2: Face recognition results vs camera distance in bright (top) and dark (bottom) lighting conditions

B. Discussion of Recognition Results

The experimental results demonstrate that, as expected, the model's performance depends on subject-camera distance and lighting conditions. In scenarios with bright lighting, the percentage of correctly recognized faces is higher compared to dark environments when using the same distance settings. This confirms that proper illumination plays a significant role in enhancing facial recognition accuracy. Additionally, recognition performance improves at closer distances, as proximity allows for more effective detection of facial features. In our experiments (Figure 2), the percentage of correct recognitions was observed to be 20.57% for bright lighting at close range, 15.97% for bright lighting at a far distance, 13.71% for low lighting at close range, and 10.41% for low lighting at a



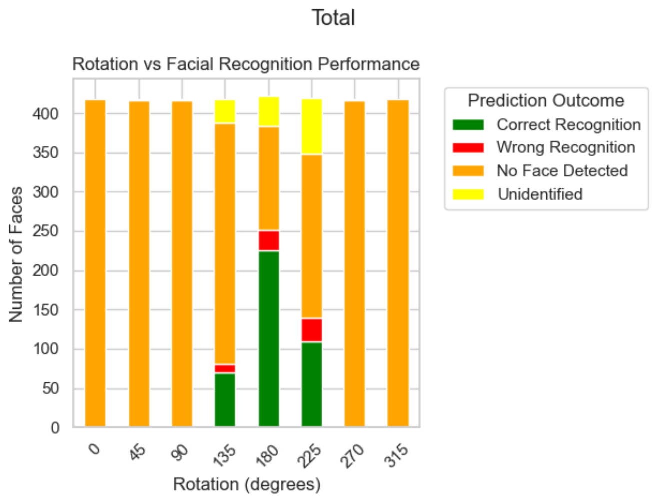


Fig. 3: Face recognition results in bright (top) and Dark (bottom) lighting conditions

far distance. With regard to camera view angles (Figure 3), the algorithm –trained on frontal (180°) faces– exhibited, as expected, the best performance at 180° (75.84% and 53.32% for bright and dark environments, respectively). Moderate ($\pm 45^0$) deviations from the frontal angle led to a noticeable decline in accuracy, which fell to 24.4% and 45.93% at 135° and 225° respectively, in bright environments. For all other off-frontal angles the (frontally-trained) model failed completely, as expected, to detect faces.

These results highlight the impact of capture conditions on face recognition performance, emphasizing the need for diverse and well-annotated datasets, like the one presented in this paper, in training robust models.

VII. Conclusion

In this paper, we presented an AR-based application designed for generating mixed (real and synthetic) human model datasets in real-world environments. The application provides an easy, scalable and flexible alternative to traditional dataset collection methods. By leveraging AR technology, it enables generation of diverse annotated datasets in a systematic way. Such datasets can be used to improve the robustness of face recognition and human detection models, or benchmark their performance. One such dataset was generated in this work.

Our experiments demonstrate the impact of key factors, such as lighting conditions and viewing angles, on face recognition accuracy. The results emphasize the necessity of diverse training data for developing models that perform reliably in real-world scenarios. Furthermore, the structured metadata exported alongside the dataset facilitate precise benchmarking and evaluation of computer vision algorithms.

The dataset and application are publicly available, encouraging further research and experimentation in the field of human-centric computer vision and, hopefully, facilitating the development of more accurate and generalizable models.

Future research will focus on diversifying humanoid models and their animations, simulating dynamic environmental changes, and applying domain adaptation techniques to enhance real-world applicability. Additionally we will explore the use of AR-generated data for training deep learning models in other computer vision tasks.

Acknowledgment

This work was funded in part from the European Union's research and innovation projects OpenDR (grant agreement no 871449, Horizon 2020) and RoboSAPIENS (grant agreement no 101133807, Horizon Europe).

References

[1] M. Wang and W. Deng, "Deep Face Recognition: A Survey," *Neurocomputing*, vol. 429, pp. 215–244, 2021.
[2] F. Liu et al., "Deep Learning Based Single Sample Face Recognition: A Survey," *Artificial Intelligence Review*, 2023.
[3] M. A. Khan, M. Mittal, L. M. Goyal, and S. Roy, "A Deep Survey on Supervised Learning Based Human Detection and Activity Classification Methods," *Multimedia Tools and Applications*, vol. 80, pp. 27867–27923, 2021.
[4] D. Schmalstieg and T. Hollerer. *Augmented Reality: Principles and Practice*. Addison-Wesley Professional, 2016.
[5] J. Kim, H. Lee, and S. Park, "Augmented Reality-Based Synthetic Data Generation for Object Detection Enhancement," *Journal of Information and Communication Convergence Engineering*, vol. 21, no. 1, pp. 98–105, 2023.
[6] C. Symeonidis, C. Anastasiadis, and N. Nikolaidis. A UAV Video Data Generation Framework for Improved Robustness of UAV Detection Methods. In *2022 IEEE 24th International Workshop on Multimedia Signal Processing (MMSP)*, pages 1–5, 2022.
[7] C. Symeonidis, P. Nousi, P. Tosidis, K. Tsampazis, N. Passalis, A. Tefas and N. Nikolaidis Efficient realistic data generation framework leveraging deep learning-based human digitization. In *International Conference on Engineering Applications of Neural Networks*, pages 271–283, 2021. Springer.
[8] Unity Technologies, *Unity*. Available online: https://unity.com/.
[9] Unity Technologies. *AR Foundation - Documentation*, Available online: https://docs.unity3d.com/Packages/com.unity.xr.arfoundation@5.0/manual/index.html
[10] Google Developers. *ARCore API Reference*, Available online: https://developers.google.com/ar/reference/
[11] Google Developers, "AR Fundamentals," 2024. Available online: https://developers.google.com/ar/develop/fundamentals.
[12] Mixamo. Available online: https://www.mixamo.com/
[13] A. Geitgey, "face_recognition: A simple face recognition library for Python," GitHub, Available: https://github.com/ageitgey/face_recognition.