Simulator for the OREO robotic head

Rajan Iyengar, University of Waterloo
Jack Zhu, University of Waterloo
Bryan Tripp, University of Waterloo
Email: srk.iyengar@gmail.com

Abstract

Eye movements are essential to human vision, and may become increasingly important in robots as their visual sophistication increases. We previously developed OREO, the first robotic head that matches the human visual system in eye movement velocity, range of motion, and stereo baseline. The programming and training of robots is greatly facilitated by simulation, but no simulator has been available for OREO so far. A useful simulator for OREO must provide realistic and varied visual input, in addition to modelling its motion physics. Here we present an OREO simulator that combines a physics model, implemented in PyBullet, with visual stimuli from AI Habitat. We briefly describe an example application and future work.

1 Introduction

A key design feature of the human visual system is that only a small region in the central visual field is analysed in detail. This requires frequent eye movements to understand a scene, and ultimately allows sophisticated visual understanding on a practical energy budget. Artificial vision on standard hardware such as GPUs is much less energy efficient than biological vision, suggesting that increasingly sophisticated artificial vision systems may benefit from a related strategy in the future.

OREO [1] shown in Fig. 1 is a robotic head that approximates key human visuo-motor capabilities through its baseline for stereo image capture and its ability to saccade, verge, and fixate. It has three degrees-of-freedom (DOF) for the neck and head movements and supports saccade-like camera movements in both yaw and pitch through two independent additional degrees-of-freedom for each eye.

Training such a physical robot is fraught with practical difficulty. Simulation is an important and standard alternative in robotics. Developing candidate solutions through simulation can reduce wear and the risk of damage, and increase the speed of conducting experiments. However, simulations that are not sufficiently realistic may not generalize well to hardware. In this paper we describe a simulation system for OREO within Habitat-Sim [2, 3] that uses PyBullet [4] to model OREO's physics. Habitat-Sim provides detailed scans of real environments that can provide sufficiently realistic visual stimuli for many purposes. We elaborate on the various parts and extensions that make up the simulation system.

2 System Design

The simulation system implements a vision capability within Habitat-Sim for OREO to traverse and view 3D scenes. It uses PyBullet to enforce the limits of OREO's movements, identify collision among OREO parts and calculate rotation quaternions for any gaze direction corresponding to the yaw and pitch angle of the eyes or the neck. Then, given 3D positions and orientations of each camera from the PyBullet model, it captures video frames from these perspectives from Habitat-Sim.

2.1 Integration with Habitat-Sim

Habitat-Sim provides egocentric visualization of large-scale 3D scene datasets. An agent with two independently rotating RGB sensors is configured within Habitat-Sim to mimic OREO's baseline. Optical parameters like sensor dimensions and field of view (FOV) angles are adjustable, allowing us to match various lenses that can be used on OREO. The two sensors can verge to fixate on the same point or gaze towards the same direction of a scene with respect to Habitat-Sim's world coordinate system (WCS). We have separated the Habitat-Sim agent rotation into two components, body rotation

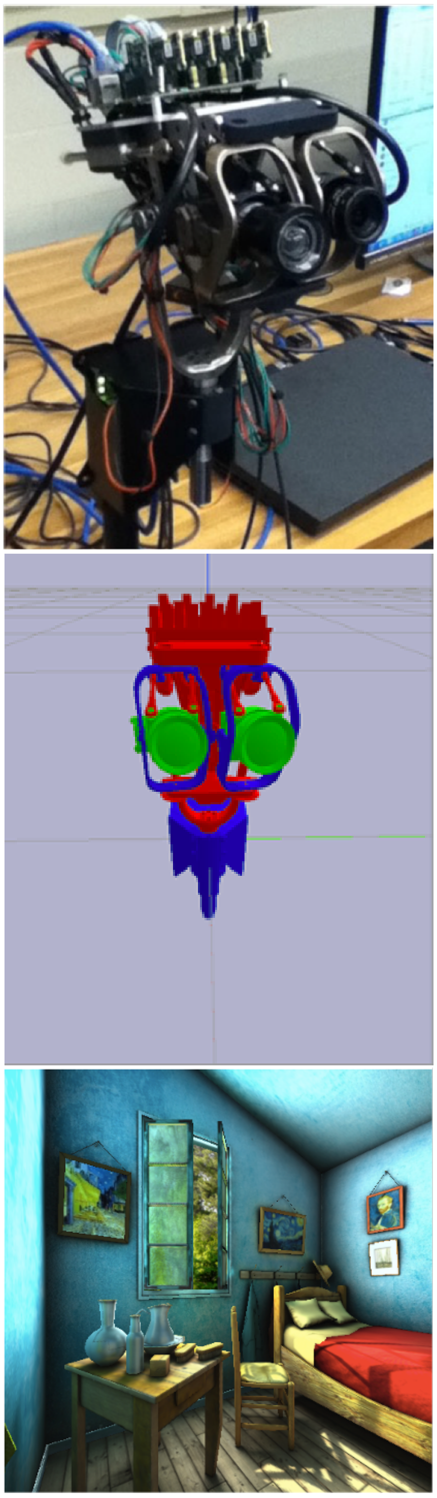

Fig. 1: Top, The physical OREO robot. Middle, Rendering of PyBullet physics model of OREO. Bottom, Example OREO-perspective image from Habitat-Sim (left eye).

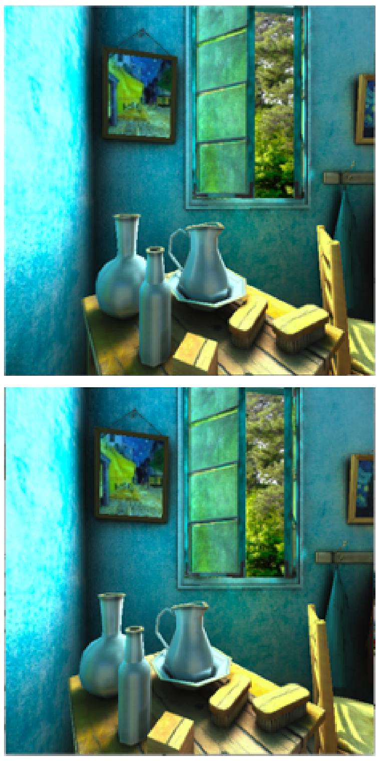

Fig. 2: Sample stereo images from the OREO model in one of the Habitat-Sim environments. Top, image from left eye. Bottom, image from right eye.

of an agent entity carrying OREO and the separate head-neck rotation of OREO. We track them separately to adhere to yaw, pitch and roll limits of OREO's head-neck rotation.

2.1.1 Coordinate Frames, Rotations, and Translation

Within the simulation system we maintain four coordinate frames. The left and right sensor frames have their origin at the center of the eye on either side of the frame for the head-neck. Head-neck rotation impacts the eye sensor positions and orientations, and the eye sensors can also rotate independently of the head. The imaginary body carrying the robot head has its own coordinate frame. The head-neck of OREO and the the body frame together determine the total rotation of the agent frame within Habitat-Sim for any movement. Moving the eye sensors amounts to aligning the gaze direction with the viewing axis of the sensors. Only the body frame can translate independently.

2.1.2 Fixation and Vergence

Each eye sensor can be made to to fixate to a 3D point or align with a gaze direction in the WCS. To mimic vergence, OREO eyes need to orient toward a common point. For a known 3D point, the unit vector connecting the point to the center of the lens provides the yaw and pitch for each eye.

2.2 PyBullet simulation and integration

The PyBullet simulation and integration is essential to ensure that the mechanical movements necessary for OREO to rotate are executed without resulting in any mechanical collision between moving parts. It also generates the rotation quaternion for any given gaze position for the different joints. We use a Unified Robot Description Format (UDRF) file representation of the geometry of OREO in 'DIRECT' connection mode with physics simulation. The limits of the degrees of freedom are imposed outside the Pybullet model by the simulation software. The yaw and pitch of each eye is controlled by two linear actuators in OREO, and we set the linear actuator positions as needed to achieve desired gaze directions.

3 Current and Future Work

We are using the simulation system to model task-free visual attention using stereo images (Fig. 2). This work only uses the kinematics of the PyBullet model. However, the PyBullet model's dynamics can also be used in combination with Habitat-Sim. Separately, the model dynamics are being used to develop an improved controller for the robot. Future extensions of this work will include a foveated rendering pipeline (with greater magnification in the image centre) to simulate an alternative foveated lens.

References

[1] S. Huber, B. Selby, and B. Tripp, "OREO: An open-hardware robotic head that supports practical saccades and accommodation," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 2640–2645, 2018.

[2] Manolis Savva*, Abhishek Kadian*, Oleksandr Maksymets*, Y. Zhao, E. Wijmans, B. Jain, J. Straub, J. Liu, V. Koltun, J. Malik, D. Parikh, and D. Batra, "Habitat: A Platform for Embodied AI Research," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

[3] A. Szot, A. Clegg, E. Undersander, E. Wijmans, Y. Zhao, J. Turner, N. Maestre, M. Mukadam, D. Chaplot, O. Maksymets, A. Gokaslan, V. Vondrus, S. Dharur, F. Meier, W. Galuba, A. Chang, Z. Kira, V. Koltun, J. Malik, M. Savva, and D. Batra, "Habitat 2.0: Training home assistants to rearrange their habitat," *arXiv preprint arXiv:2106.14405*, 2021.

[4] E. Coumans and Y. Bai, "Pybullet, a python module for physics simulation for games, robotics and machine learning," 2016.