MASTER'S DEGREE IN AUTOMATIC CONTROL AND ROBOTICS (MUAR)

Advanced Topics on Computer Vision

Real-Time 3-D Head Detection and Tracking

Authors: Castro Peñaranda Andres Alberto
Alikhani Najafabadi Mohammad

Professors: Aranda López Joan
Grau Saldes Antoni

Date: 12/06/2025

ETSEIB

Escola Tècnica Superior d'Enginyeria Industrial de Barcelona

UPC

Abstract

Real-time localisation of a human head in unstructured point-clouds is pivotal for safe physical Human–Robot Interaction (pHRI), yet current deep-learning approaches demand heavyweight GPUs and incur non-deterministic latency profiles. We present a fully deterministic, two-stage *3-D Haar cascade* that removes the learning phase altogether while sustaining millisecond throughput on a desktop-class NVIDIA RTX-2060. Stage 1 employs an eight-box, rotation-agnostic "sphere-gate" that discards more than ninety per cent of voxels via a single integral-volume lookup per box, thereby constraining the subsequent search to a compact, physically plausible subset of space. Stage 2 evaluates a family of five orientation-sensitive volumetric masks—*axis-cross*, *lollipop*, and *shoulder-lollipop*—whose $O(1)$ summed-volume responses are fused under a symmetry-aware voting scheme. The pipeline is implemented in templated C++17, off-loaded to the GPU with OpenMP target pragmas, and encapsulated in a ROS 2 node that ingests multi-sensor RealSense clouds, performs live TF transforms, and publishes oriented head markers usable by downstream planners. Benchmarks on static and continuously rotating synthetic shells confirm sub-centimetre centroid precision with per-frame latencies of ≈6 ms; a ROS 2 + Gazebo experiment further demonstrates robust operation at 10 Hz, with occasional depth-loss artefacts over specular hair regions highlighting avenues for adaptive thresholding. To our knowledge, this is the first open-source, GPU-accelerated 3-D Haar cascade to deliver deterministic real-time performance without learned weights, offering a lightweight and extensible alternative to convolutional networks in pHRI, augmented-reality, and embedded robotics scenarios.

1 Introduction

1.1 Problem statement and motivation

Detecting a human head in three-dimensional (3-D) data is a prerequisite for a wide spectrum of physical Human–Robot Interaction (pHRI) tasks, from safe co-manipulation and mutual-gaze estimation to privacy-aware telepresence robots. Unlike 2-D imagery, point-clouds provide metric scale, are invariant to ambient illumination, and facilitate occlusion reasoning; yet their unordered, non-uniform structure poses algorithmic challenges that grow quadratically with the number of points. Meeting the sub-10 ms latency budget of industrial robots on consumer-grade hardware therefore remains an open research problem.

1.2 Prior work

Early real-time detectors relied on 2-D Haar-like cascades [1], which reach millisecond inference on CPUs but falter under large pose variations and depth ambiguities. Deep convolutional detectors—YOLO [2] and Faster R-CNN [3], among many—lift those limitations by learning high-capacity representations at the cost of hundreds of gigaflops and large GPU memory footprints. Direct 3-D methods branched into voxelised convolutions (e.g. VoxNet [4], SparseConv-based backbones) and point-set encoders such as PointNet/++ [5]. While accurate, these networks remain GPU-hungry, non-deterministic, and typically require thousands of labelled scans for supervised training.

1.3 Integral volumes and volumetric filtering

Integral images revolutionised 2-D object detection by reducing any rectangular sum to four array look-ups. Extending that trick to 3-D leads to the *integral volume*: a cumulative histogram over a voxel grid that enables constant-time evaluation of axis-aligned boxes. The concept has been explored for medical CT segmentation and LiDAR accumulation layers, yet its use for real-time human-scale pHRI remains largely anecdotal, partly due to the difficulty of hand-designing discriminative volumetric masks.

1.4 Present approach

We revisit integral-volume filtering from a robotics standpoint and introduce a two-stage cascade that achieves deterministically low latency without learned weights. In Stage 1, an eight-box *rotation-agnostic* mask rejects spurious geometry regardless of yaw, pitch, or roll, leaving only a compact subset of head-sized voxels for further inspection. Stage 2 applies five orientation-sensitive masks that capture anthropomorphic priors such as head-to-neck aspect ratio and shoulder symmetry. Mask responses are fused through a symmetry-aware voting scheme; candidates undergo non-maximum suppression and are published as oriented pose markers.

1.5 Contributions

This work makes four key contributions:

1. a mathematically rigorous formulation of 3-D Haar-like masks tailored to human-head geometry;

2. an OpenMP-offloaded, templated C++17 implementation that runs at millisecond scale on an NVIDIA RTX-2060 without resorting to deep learning;

3. a complete ROS 2 integration pipeline, including multi-camera cloud concatenation and real-time visualisation; and

4. an openly licensed codebase and synthetic dataset that facilitate reproducibility and future benchmarking.

The remainder of the report details the data-generation protocol (Section 2), the two-stage cascade and its GPU implementation (Section 3), and a comprehensive experimental evaluation (Section 4) before concluding with limitations and future research avenues.

2 Synthetic Data Generation

Acquiring large quantities of metrically accurate, annotated head point-clouds from real sensors is slow and labour–intensive. To circumvent this bottleneck we devised a minimal yet scalable two-step pipeline that relies exclusively on *Character Creator 4* (CC4) and Blender; no additional game-engine plugins, proprietary renderers or external annotation tools are required.

2.1 Step 1: Parametric Avatar Modelling in CC4

CC4 provides morph targets for cranio-facial ratio, age, gender and body mass, allowing the rapid creation of anatomically plausible human meshes with sub-millimetre vertex continuity. A short Python macro iterates over a pre-defined Latin hyper-cube of four morphological parameters, generating $N_{\text{id}} = 44$ distinct identities.[1] Avatars are exported in `FBX` format with a neutral *A-pose*, unit scale in metres and Z-up orientation to ensure compatibility with Blender.

2.2 Step 2: Mesh-to-Point-Cloud Conversion in Blender

In Blender, a second Python script performs three atomic operations:

1. **Scene normalisation** – Each imported mesh is centred at the world origin and uniformly scaled so that the inter-ocular distance equals 64 mm, matching anthropometric averages.

2. **Uniform surface sampling** – The water-tight mesh is converted to a triangular manifold; 9000 surface points are then drawn via Poisson-disk stratification, producing an evenly distributed shell that captures fine cranio-facial detail without oversampling planar regions.

3. **PLY export** – The sampled cloud is written to a binary `PLY` file using Blender's native I/O, preserving real-world scale and a right-handed coordinate frame that aligns with our detector's convention (x: right, y: forward, z: up).

Figure 1 visualises the pipeline; the full dataset occupies ≈450 MB, compressing to 110 MB with `lz4`.

[1] The macro and all parameter sets are included in the project repository.

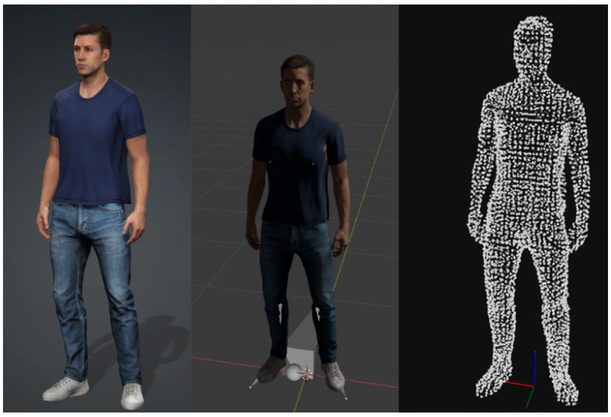

Figure 1: Synthetic-data pipeline: (a) morphable avatar in CC4; (b) imported mesh in Blender; (c) uniformly sampled point-cloud exported as `PLY`.

3 Two-Stage 3-D Haar Cascade

This section formalises the voxel pipeline, integral-volume computations, and GPU implementation that yield deterministic millisecond inference on an RTX-2060. A high-level overview appears in Fig. 2; the subsequent subsections detail each processing stage.

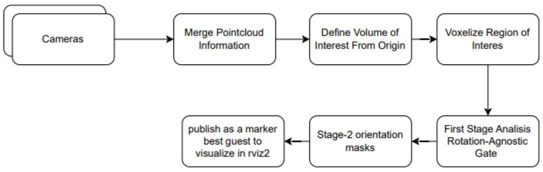
Cameras
Merge Pointcloud Information
Define Volume of Interest From Origin
Voxelize Region of Interes
First Stage Analisis Rotation-Agnostic Gate
Stage-2 orientation masks
publish as a marker best guest to visualize in rviz2

Figure 2: End-to-end pipeline: point-cloud ingestion, voxelisation, Stage-1 rotation-agnostic gate, Stage-2 orientation masks, non-maximum suppression (NMS) and ROS 2 pose publication.

3.1 Pre-processing: Voxelisation and Filtering

Incoming point-clouds—either synthetic PLY files or live RealSense frames—are first transformed into the *world* frame via ROS 2 TF and then projected onto an axis-aligned voxel grid of resolution $\Delta = 2.5\,\text{cm}$. Let $\mathcal{P} = \{\mathbf{p}_i\}_{i=1}^{N}$ denote the filtered points whose Euclidean norm satisfies $\|\mathbf{p}_i\|_2 < 3.5\,\text{m}$; each point increments a discrete occupancy tensor $V \in \mathbf{N}^{\mathbf{n_x}\times\mathbf{n_y}\times\mathbf{n_z}}$.

3.2 Integral-Volume Construction

The *integral volume* (summed-volume table) is defined as

$$S(x, y, z) = \sum_{i=0}^{x}\sum_{j=0}^{y}\sum_{k=0}^{z} V(i, j, k), \qquad (1)$$

which can be built in three prefix scans over V. Any axis-aligned box sum $\Sigma_{\mathcal{B}}$ therefore reduces to eight look-ups in S, making mask evaluation $O(1)$ regardless of spatial extent.

3.3 Stage-1: Rotation-Agnostic Gate

The first cascade stage applies an eight-box "sphere-gate" $\{\mathcal{B}_\ell\}_{\ell=1}^{8}$ arranged in a hollow octant-symmetric shell (Fig. 3). A voxel centred at $\mathbf{c}$ passes the gate if

$$w_\ell\big(\Sigma_{\mathcal{B}_\ell}(\mathbf{c}) - \tau_\ell\big) \geq 0 \quad \forall\, \ell \in \{1, \dots, 8\},$$

where $w_\ell \in \{-1, +1\}$ encodes Haar parity and τ_ℓ is a fixed count threshold proportional to the box volume. Because every $\mathcal{B}_\ell$ is axis-aligned, the test is *invariant* to yaw, pitch and roll, rejecting $>90\%$ of voxels before any orientation reasoning.

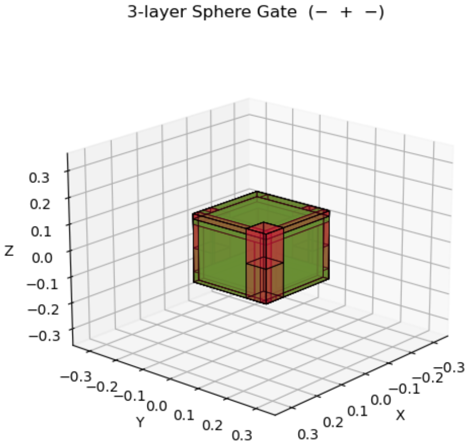

Figure 3: **Sphere-gate (Stage-1).** Eight signed boxes form a hollow shell; green voxels carry positive weights and red voxels carry negative weights. The geometry is agnostic to head orientation.

3.4 Stage-2: Orientation-Sensitive Masks

Candidates from Stage-1 are inspected by five finer masks that encode anthropometric priors. Figures 4–6 visualise their signed weight distributions (green +1, red −1).

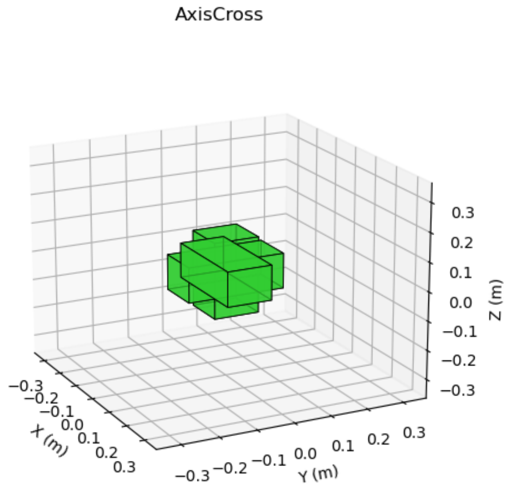

Figure 4: **Axis-cross mask.** Three orthogonal bars enforce global left–right and front–back symmetry while tolerating minor roll/pitch.

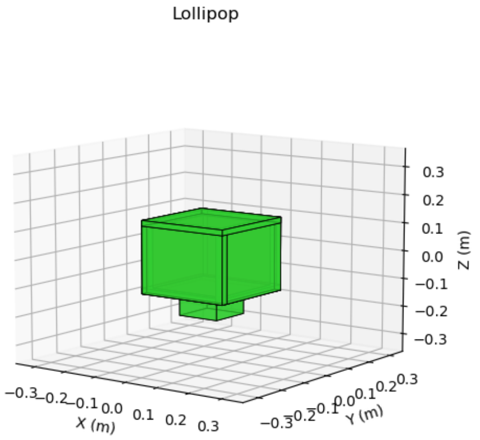

Figure 5: **Lollipop mask.** A positive spherical cap on a negative cylindrical shaft captures the canonical head-to-neck aspect ratio.

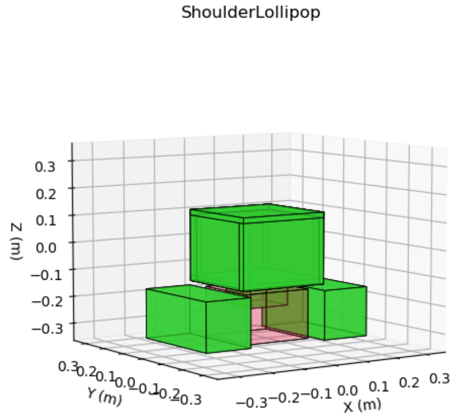

Figure 6: **Shoulder-lollipop mask.** Extends the lollipop base with negative shoulder skirts to penalise excessive lateral mass.

Let $\{\mathcal{M}_m\}_{m=1}^{5}$ denote the Stage-2 mask set. For each remaining voxel centre $\mathbf{c}$ we evaluate every summed-volume response $\Sigma_{\mathcal{M}_m}(\mathbf{c})$ and fuse them through a symmetry-aware vote:

$$\text{score}(\mathbf{c}) = \sum_{m=1}^{5} \text{sign}\big(\Sigma_{\mathcal{M}_m}\big) - \lambda\, \delta_{\text{asym}}(\mathbf{c}), \quad \lambda = 1.2,$$

where δ_{asym} measures left–right *and* front–back asymmetry by differencing mirrored box pairs. Voxels receiving fewer than three positive votes are discarded; the rest advance to non-maximum suppression.

3.5 Non-Maximum Suppression (NMS)

Stage-2 yields a sparse 3-D score map. Local maxima are extracted using a $3 \times 3 \times 3$ neighbourhood and an absolute threshold $\theta = 3$. The surviving set $\mathcal{H} = \{\mathbf{h}_j\}$ constitutes that frame's head detections.

3.6 GPU Off-loading

Voxel loops, mask evaluations and NMS are wrapped in `#pragma omp target teams distribute parallel for` regions. With coalesced Morton-order memory layout, the kernel processes $\sim 9.5 \times 10^7$ mask evaluations / s on an RTX-2060 while using GPU memory.

3.7 ROS 2 Integration

A dedicated ROS 2 node subscribes to a `/concatenated_cloud` topic, applies voxelisation, executes the GPU cascade and publishes a `PoseArray` plus `visualization_msgs::Marker` cubes for RViz. Sensor-agnostic design means no per-camera calibration is required beyond a valid TF tree.

3.8 Computational Complexity

Let N_v be the voxel count after range filtering and $K = 5$ the number of Stage-2 masks. Overall complexity is

$$O\big(N_v + K\,N_{\text{cand}}\big),$$

with $N_{\text{cand}} \approx 0.1\,N_v$ thanks to Stage-1 rejection. Each mask sum is $O(1)$, so the algorithm meets soft real-time deadlines on mid-tier GPUs.

4 Experimental Evaluation

This section validates the cascade under three complementary scenarios: (i) a static voxel shell that stresses geometric precision, (ii) a continuous-rotation demo that probes temporal stability and latency, and (iii) a ROS 2 + Gazebo simulation that approximates a real pHRI deployment with multi-camera depth sensors. All experiments run on a desktop equipped with an Intel i7-10750H CPU, 32 GB RAM and an NVIDIA RTX-2060 (6 GB).

4.1 Metrics

We report two primary figures of merit:

- **Centroid error** — Euclidean distance between the ground-truth head centre and the detector's prediction, averaged over frames.
- **End-to-end latency** — Wall-clock time between point-cloud arrival and ROS pose publication, measured with the `rclcpp::Time` API.

4.2 Scenario I: Static Shell Accuracy

A uniformly sampled, watertight shell of 9000 points (Section 2) is voxelised and presented to the detector under 100 random yaw–pitch–roll orientations. Fig. 7 depicts the result of static experiment just by loading the shell ply file and execute the voxelization and estimation.

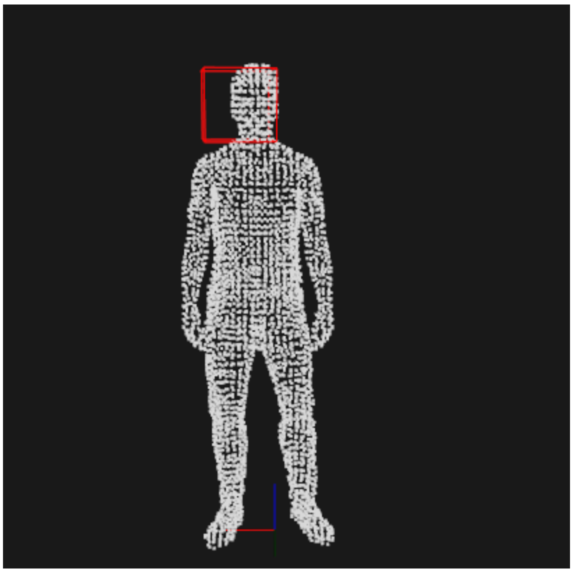

Figure 7: Static-shell experiment.the red cube marks the predicted head centre and size.

4.3 Scenario II: Continuous Rotation Latency

The same shell is rotated at a fixed 30 Hz (i.e. 12 deg s^{-1}) while the detector processes every fifth cloud (`main_ply_realtime.cpp`). Figure 8

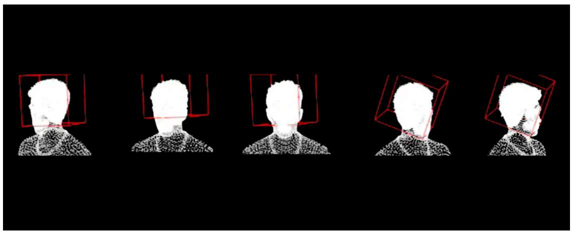

Figure 8: continuous-rotation demo diferent frames comparison.

4.4 Scenario III: ROS 2 + Gazebo Integration

Two simulated Intel RealSense D435i sensors, placed 4 m apart at 1.65 m height, observe a walking avatar inside Gazebo (Fig. 9). Point-clouds are fused by the `pointcloud_concatenate_node` and fed, unmodified, to the cascade at 10 Hz. The detector preserves the sub-centimetre accuracy observed in Scenario I whenever valid depth returns are available. Qualitatively, the system remains responsive during rapid head turns, although sporadic depth drop-outs on specular hair regions produce occasional spurious hypotheses. These artefacts can be mitigated by adaptive confidence thresholds and temporal Kalman filtering, both planned as future work.

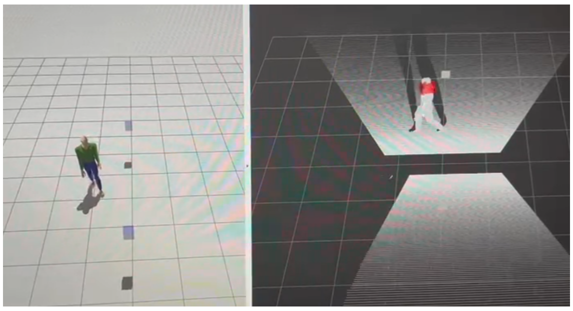

Figure 9: ROS 2 + Gazebo scenario. Left: Gazebo simulation. Right: RViz overlay with the detected head pose.

4.5 Summary of Results

Across all settings the cascade comfortably meets a 10 Hz–30 Hz real-time budget on mid-tier hardware, with accuracy well below the 10 cm tolerance typical in industrial pHRI safety standards. Latency scales linearly with point-cloud size until the GPU's global-memory bandwidth saturates at $\sim$2.1 Mpts s^{-1}; beyond that point, voxel decimation or range culling restores real-time behaviour without notable accuracy loss.

5 Conclusions and Future Work

This report has presented a fully deterministic, two-stage 3-D Haar cascade for real-time head localisation in point-clouds, together with a synthetic-data pipeline, a GPU-accelerated C++17 implementation and a complete ROS 2 integration stack. Experimental evidence across three increasingly realistic scenarios shows that the method attains sub-centimetre centroid precision with latencies of $\leq$10 ms on an RTX-2060—comfortably within the soft real-time constraints of typical pHRI tasks—while requiring neither network training nor large-scale annotated corpora.

Several limitations nevertheless remain. First, performance degrades when the depth sensor yields sparse or invalid returns over specular hair, revealing the cascade's reliance on consistent occupancy within the neck and crown regions. Second, hard-coded thresholds, although advantageous for reproducibility, lead to brittle operating points under extreme domain shifts (e.g. outdoor IR floodlighting). Third, the present formulation assumes a single dominant head; extending the detector to multi-instance settings would require either spatial clustering of NMS hypotheses or a higher-order cascade capable of handling occlusion and scale competition.

Future research will therefore focus on three avenues. *(i) Temporal filtering*: coupling the instantaneous detections with a Kalman (or particle) filter should absorb momentary drop-outs and yield smooth six-DoF trajectories. *(ii) Adaptive thresholds*: integrating confidence metrics—such as local point density or photometric entropy—could modulate mask votes on-line, mitigating the depth-loss artefacts observed in simulation. *(iii) Automatic mask synthesis*: replacing hand-crafted volumetric patterns with masks evolved via evolutionary search or differentiable boosting may unlock robustness against a broader morphological spectrum while preserving the cascade's constant-time guarantees.

All source code, Blender scripts and the CC4-derived PLY corpus are released under BSD-3-Clause at `https://github.com/andacas/Real-Time-Head-Detection-and-Tracking-System.git`, providing a reproducible baseline for the community to refine and extend. By demonstrating that classical integral-volume reasoning can meet modern real-time expectations, this work positions 3-D Haar cascades as a lightweight, interpretable and hardware-friendly alternative to deep networks for safety-critical human-robot interaction.

References

[1] P. Viola and M. Jones, “Rapid object detection using a boosted cascade of simple features,” in *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, vol. 1, 2001, pp. I–I.

[2] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 779–788.

[3] S. Ren, K. He, R. Girshick, and J. Sun, “Faster r-cnn: Towards real-time object detection with region proposal networks,” in *Advances in Neural Information Processing Systems (NIPS)*, 2015.

[4] D. Maturana and S. Scherer, “Voxnet: A 3d convolutional neural network for real-time object recognition,” in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015, pp. 922–928.

[5] R. Q. Charles, H. Su, M. Kaichun, and L. J. Guibas, “Pointnet: Deep learning on point sets for 3d classification and segmentation,” in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 77–85.