Active Vision for Social Navigation

Jack Vice and Gita Sukthankar *

Posted Date: 2 April 2026

doi: 10.20944/preprints202604.0068.v1

Keywords: social navigation; active vision; reinforcement learning

 doi:10.20944/preprints202604.0068.v1

Article

Active Vision for Social Navigation

Jack Vice, Gita Sukthankar *

Department of Computer Science, University of Central Florida, Orlando, FL 32816, USA
* Correspondence: gita.sukthankar@ucf.edu

Abstract

Traditional social navigation systems often treat perception and motion as decoupled tasks, leading to reactive behaviors and perceptual surprise due to limited field of view. While active vision—the ability to choose where to look—offers a solution, most existing frameworks decouple sensing from execution to simplify the learning process. This article introduces a novel joint reinforcement learning (RL) framework (**A**ctive **V**ision for **S**ocial **N**avigation) that unifies locomotion and discrete gaze control within a single, end-to-end policy. Unlike existing factored approaches, our method leverages a model-based RL architecture with a latent world model to explicitly address the credit assignment problem inherent in active sensing. Experimental results in cluttered, dynamic environments demonstrate that our joint policy outperforms factored sensing-action approaches by prioritizing viewpoints specifically relevant to social safety, such as checking blind spots and tracking human trajectories. Our findings suggest that tight sensorimotor coupling is essential for reducing perceptual surprise and ensuring safe, socially aware navigation in unstructured spaces.

Keywords: social navigation; active vision; reinforcement learning

1. Introduction

A central limitation of the social navigation pipeline is often the narrow instantaneous field of view (FOV) of a forward-facing camera. In cluttered, occluded, or dynamic scenes, pedestrians can enter the robot's path from oblique angles without being observed until they are already dangerously close, reducing the effectiveness of downstream prediction and increasing the likelihood of high-risk encounters. Active vision addresses this limitation by treating viewpoint selection as a policy in which the robot learns to choose where to look while simultaneously controlling locomotion. Concretely, active vision improves social navigation with a discrete gaze state over a small set of yaw-offset views, enabling proactive sensing behaviors such as checking blind spots and following human trajectories that reduce perceptual surprise and improve social navigation in unstructured environments.

This article introduces a new reinforcement learning framework: **A**ctive **V**ision for **S**ocial **N**avigation. Our proposed method augments a perception–prediction–control pipeline with 1) a discrete view-selection mechanism, 2) joint reinforcement learning over locomotion and gaze actions, and 3) a spatiotemporal attention mechanism to predict near-future human occupancy. Each control step consists of three stages: selecting a gaze window index (pan) together with a wheel-motion command; rectifying the selected view and computing compact perceptual channels as the state observation; and providing the RL agent with a fused low-dimensional observation that includes both geometry and semantic cues alongside predicted human occupancies.

Learning viewpoint control and navigation control simultaneously is substantially harder than learning either in isolation, and much of the active-vision literature explicitly avoids this fully coupled formulation. Prior work typically reduces complexity by factorizing sensing and task behavior into separate policies or asynchronous loops, thereby shortening credit-assignment paths and stabilizing learning. Shang and Ryoo formulate active vision with separate sensory and motor policies to address limited observability and indirect supervision for viewpoint control [1]. Dass et al. show that separating

 doi:10.20944/preprints202604.0068.v1

information-seeking from task execution makes learning more tractable by assigning distinct roles to context acquisition and downstream decision making [2], while Wang et al. explicitly decouple observation and action in an asynchronous active vision–action framework [3]. Even self-supervised approaches such as [4] sidestep the coupling by first rewarding camera behaviors that improve predictability, rather than requiring task reward to supervise sensing choices directly.

Our formulation learns a single joint policy over the product action space (navigation $\times$ view angle), forcing the agent and world model to solve coupled credit assignment. This is challenging for three reasons: gaze actions have delayed utility since a pan action improves *future* observation quality rather than yielding immediate reward; gradients compete, as early navigation mistakes dominate returns while the signal for beneficial gaze shifts is weak and indirect; and view transitions expand the learned dynamics, changing the observation distribution even when the robot does not move. Despite this difficulty, joint optimization is well-motivated here because viewpoint control is not an auxiliary behavior but a task-dependent component of navigation itself, where the robot looks directly determines what it can infer about nearby humans and traversable space, which in turn shapes the optimal motion decision. Factored approaches [1–3] risk biasing the sensing strategy toward broadly informative views rather than those specifically useful for socially safe locomotion, whereas a joint policy can learn task-specific sensorimotor coordination such as briefly redirecting gaze to resolve an ambiguous human interaction before choosing a safer path. This long-horizon coupling is precisely what model-based RL is designed to capture through imagined trajectories over a learned latent world model [5].

For this to work, the world model must learn to predict how viewpoint changes alter subsequent observations. Because gaze and motor commands are jointly sampled and modeled, the latent world model associates pan actions with their resulting visual consequences, when the agent pans toward a pedestrian, the world model links that action to the ensuing observation change and any downstream proxemic penalty. Imagined rollouts can therefore assign higher value to action sequences that proactively redirect gaze toward anticipated human occupancy, producing look-before-you-drive behavior through temporal credit assignment.

Section 2 provides an overview of related work on social navigation. Our proposed architecture is described in Section 3. Section 4 presents a comparison of our method against several competitive benchmarks in three social navigation scenarios. Our results show that active vision produces a clear statistical difference in performance at achieving a higher number of navigation goals with fewer near collisions.

2. Related Work

Social navigation, the ability for robots to navigate safely and predictably in environments shared with humans—has evolved significantly with advances in perception, learning, and planning. Early work on social robot navigation emphasized proxemics and safety-aware planning [6,7], while recent trends leverage deep learning for trajectory forecasting and policy learning. Social navigation requires autonomous robots to reason not only about geometric constraints but also about human social behaviors, such as maintaining interpersonal distance and yielding right-of-way. Hall's proxemics theory formalized the interpersonal zones that robots must respect [8]; this article evaluates the performance of the RL agent using Hall's suggested proxemic zones for personal and social distances. Early planners extended the Dynamic World Approach (DWA) and velocity-obstacle schemes with proxemic costs to remain inside socially acceptable regions [9–12]. These rule-based methods guarantee safety but struggle with the diversity of human behaviors in dense crowds.

End-to-end learning replaces hand-tuned costs with policies optimized directly for social compliance. Tai et al. used GAIL to learn depth-only navigation that implicitly captures human preferences [13]. DRL-VO merges RL with velocity-obstacle geometry for faster convergence in cluttered scenes [14], while uncertainty-aware RL penalizes high-variance actions to curb the "freezing" phenomenon [15]. Reward shaping based on heuristics or social norms further steers policies toward

 doi:10.20944/preprints202604.0068.v1

courteous behavior [16]. Our proposed method is unique in its inclusion of active vision to prevent perceptual surprise and its ability to learn joint policies for sensing and action.

3. Materials and Methods

Our system architecture is a multi-process pipeline that fuses human intent prediction with terrain-aware reinforcement learning. For each new camera frame, a YOLOv11 detector identifies current pedestrian locations while a spatiotemporal attention mechanism predicts future human occupancy patterns from temporal frame sequences. Concurrently, monocular depth estimation extracts geometric scene structure from the latest RGB frame. These three information streams—current pedestrian masks overlaid on grayscale imagery, attention-based trajectory predictions, and depth maps—are fused into a compact 96×96×3 observation tensor that encodes both social and geometric constraints. This multimodal representation serves as input to a DreamerV3 [5] reinforcement learning agent that learns socially compliant navigation policies through latent world modeling, enabling real-time decision making that respects both human proxemics and terrain traversability in unstructured environments. Figure 1 shows the active vision pipeline in which a single wide-angle fisheye camera provides five discrete view frames. The RL agent learns a joint policy with motor commands and view selection.

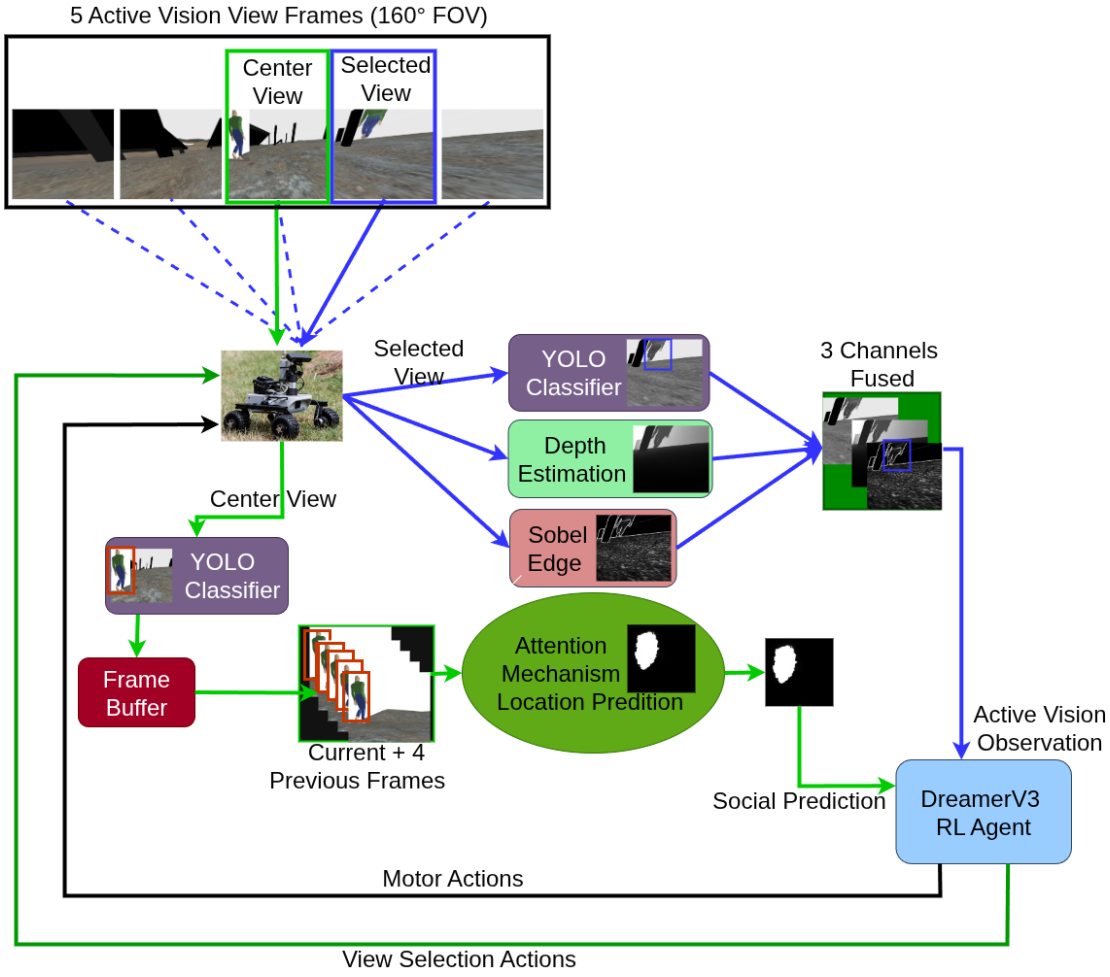

Figure 1. Active vision social-navigation pipeline. A wide-angle fisheye camera provides five discrete view frames spanning a 160° field of view. At each step the RL agent outputs both motor commands and a view-selection action. The selected view is rectified and processed in parallel by YOLO person detection, monocular depth estimation, and Sobel edge extraction; these three outputs are stacked into a compact 3-channel fused image used for control. In parallel, the center view feeds YOLO bounding boxes into a temporal buffer that is passed to the spatiotemporal attention mechanism, which produces a social-prediction observation for the agent. This design decouples temporally consistent social prediction from active vision controlled viewpoint perception while learning joint gaze and locomotion policies end-to-end.

 doi:10.20944/preprints202604.0068.v1

3.1. Datasets

We employ the SiT Social-Interactive Trajectory (SiT) [17] corpus augmented with sequences collected in the DuNE–Gazebo [18] simulator to ensure coverage of both real and synthetic unstructured scenes, totaling just over 100k images from multiple video sequences. For every video sequence, frames are first subsampled at 10 Hz and assembled into fixed–length windows that respect the temporal offsets in $\mathcal{T}_{\text{past}} = \{0, -0.5, -1.0, -1.5, -2.0\}$ s and a prediction horizon $\tau_f = +2.0$ s. Each window thus contains $T = 5$ RGB frames (current + 4 previous) $\{\mathbf{I}_{t+k}\}_{k\in\mathcal{T}_{\text{past}}}$. A lightweight YOLOv11n model detects pedestrians in all frames; detections are linked across time via a distance–based tracker to yield short trajectories. From the centroids $\mathbf{p}_i = (x_i, y_i)$ of pedestrians in the future frame we generate a heatmap

$$H(x,y) = \max_i \exp\Big[-\tfrac{\|\mathbf{p}_i-(x,y)\|_2^2}{2\sigma^2}\Big], \qquad \sigma = 16 \text{ px}, \tag{1}$$

which is normalized to $[0, 1]$. Simultaneously we rasterize pedestrian YOLO center boxes into binary masks $\mathbf{M}_{t+k}$ and compute monocular depth $\mathbf{D}_{t+k}$ with DEPTH-ANYTHING V2 [19]. All tensors are resized to 320×320, stored as half-precision *mem-maps*, and streamed in constant-time batches during training, eliminating GPU out-of-memory crashes when scaling to the full ($\approx$ 107,000) image dataset.

3.2. Spatiotemporal Attention Mechanism

We predict near–future human occupancy as a dense heatmap from a short RGB history and per–frame person masks. At each step we form a sequence $\{(\mathbf{I}_{t-k}, \mathbf{M}_{t-k})\}_{k=0}^{T-1}$ of T RGB frames and their binary masks (from YOLO detections). Each pair is encoded by lightweight CNNs with shared structure: strided convolutions downsample from 320×320 to 80×80 and 40×40, producing a bottleneck feature $\mathbf{F}_{t-k} \in \mathbb{R}^{40\times40\times C}$ and skip maps at 80×80 and 40×40 for decoding. We then concatenate RGB and mask features channel-wise, stack them across the T timesteps, and reshape to a token sequence of length $T\cdot40\cdot40$:

$$\mathbf{Z} \in \mathbb{R}^{B\times(T\cdot40\cdot40)\times C}. \tag{2}$$

A linear layer projects $\mathbf{Z}$ to an embedding dimension D (configurable; defaults provided in code), followed by LayerNorm and the addition of a learned spatiotemporal positional embedding of the same shape. This absolute embedding lets the model encode both spatial indices and relative time implicitly.

We apply a single `SelfAttention` block with 8 heads over the token axis. The attention is the standard scaled dot-product:

$$\text{Attention}(Q,K,V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right)V, \tag{3}$$

implemented inside multi-head projections so each head attends over the D/h-dimensional subspace. The attended sequence is added residually and normalized, then passed through a linear projection and dropout. We then reshape back to $[B, T, 40, 40, D]$ and perform temporal mean pooling to obtain a single 40×40 latent per batch.

Decoding starts from the pooled 40×40 latent. We concatenate the 40×40 skip feature via a 1×1 projection and upsample with transposed convolutions to 80×80 and finally 320×320, concatenating the 80×80 skip along the way. A final 3×3 convolution with sigmoid produces the occupancy heatmap $\hat{H} \in [0, 1]^{320\times320}$.

Training targets H are built from *future* YOLO detections by rasterizing each person into an elliptical Gaussian centered at the detection centroid, with per-axis standard deviations scaled by the bounding-box size. The model is optimized with a balanced *focal+Dice* objective:

$$\mathcal{L} = \mathcal{L}_{\text{focal}}(\hat{H}, H) + \mathcal{L}_{\text{dice}}(\hat{H}, H), \tag{4}$$

where the focal term uses the usual (α, γ) parameters and the Dice term promotes overlap with soft targets (both implemented in code). For monitoring, we additionally report RMSE over the heatmap.

 doi:10.20944/preprints202604.0068.v1

We train with Adam (Optax) and gradient clipping (global ℓ_2 norm 1.0). The learning rate follows a warmup+cosine schedule (linear warmup then decay), with default mini-batches of 8 sequences and sequence length T=5 (all configurable). The model and optimizer are instantiated in JAX/Flax; training loops include on-the-fly validation and optional TensorBoard logging. Checkpoints save the full parameter PyTree and config for resumption.

3.3. Reinforcement Learning

We build on DreamerV3 [5], which learns a compact latent-space dynamics model from pixels and proprioception, and then performs 'imagined' rollouts in that latent space to train the policy and estimate long-horizon returns to optimize an actor–critic policy before executing in the environment. Specifically, DreamerV3 learns a recurrent state-space model (RSSM) whose latent evolves stochastically $z_t \sim p_\phi(z_t \mid z_{t-1}, a_{t-1})$ while reconstructing observations through a decoder $o_t \sim p_\phi(o_t \mid h_t, z_t)$. Let h_t denote the deterministic recurrent state (e.g., GRU hidden), z_t the stochastic latent, and $s_t := [h_t, z_t]$ the concatenated latent consumed by the actor, critic, and decoders. Actor and critic are then trained on imagined sequences $\{s_t\}_{t=1}^{T}$, enabling sample-efficient control from high-dimensional inputs and reducing sensitivity to pixel noise. In our navigation setting, a fused encoder (RGB + attention + depth) supplies the world model with traversability and goal cues; the learned dynamics support long-horizon, goal-directed planning that captures both skid-steer kinematics and terrain structure.

The attention-based prediction module injects explicit social priors into long-horizon planning. Let $\mathcal{O}_{t-\tau:t}$ denote the past τ seconds of observations and $\hat{\mathbf{p}}_{t+\delta} = f_\theta(\mathcal{O}_{t-\tau:t})$ the predicted pedestrian positions δ seconds ahead. These forecasts are fused into the agent's representation (as an additional observation channel and latent feature), so DreamerV3's imagined rollouts evaluate candidate actions under anticipated human occupancy rather than relying solely on raw pixels. Practically, this biases both the learned world model and the agent toward trajectories that steer around predicted human locations and sustain comfortable separation.

3.3.1. RL Agent Reward Design

The reward function is designed to balance efficient goal-directed navigation with socially compliant behavior in dynamic, unstructured environments. At each timestep t, the agent receives a composite reward R_t that encourages progress toward the navigation target while discouraging unsafe or inefficient behaviors. The reward is computed as:

$$R_t = \alpha \Delta d_t + \beta\left(1 - \frac{|\theta_t|}{\pi}\right) + \gamma v_t - \delta|\omega_t| - \epsilon - \sum_{i=1}^{N} P(d_i) + \kappa \Delta d_t \mathbf{1}_{\text{aligned}} + R_{\text{goal}}\mathbf{1}_{\text{success}} \tag{5}$$

where $\Delta d_t = d_{t-1} - d_t$ represents the decrease in Euclidean distance to the goal (positive for progress), $\theta_t \in [-\pi, \pi]$ is the heading difference between the robot's current orientation and the target direction, v_t is the linear velocity, and ω_t is the angular velocity. Social safety is incorporated through a proxemic penalty function $P(d_i)$ applied to each of N nearby human actors at distance d_i. Based on Hall's proxemic zones [8], the penalty grows in discrete stages:

$$P(d) = \begin{cases} 25.0 & \text{if } d < 0.5\text{m} \quad \text{(intimate zone)} \\ 8.0 & \text{if } 0.5\text{m} \leq d < 0.8\text{m} \quad \text{(personal zone)} \\ 2.0 & \text{if } 0.8\text{m} \leq d < 1.2\text{m} \quad \text{(social zone)} \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

The indicator function $\mathbf{1}_{\text{aligned}}$ provides an alignment bonus when the robot makes forward progress ($\Delta d_t > 0$) while well-aligned with the target ($|\theta_t| < \pi/4$). The goal reward $R_{\text{goal}} = 100.0$ is achieved when the robot reaches within 0.3m of the goal . The hyperparameters are set to $\alpha = 2.0$ (distance progress), $\beta = 0.03$ (heading alignment), $\gamma = 0.015$ (velocity), $\delta = 1.8$ (spin penalty),

 doi:10.20944/preprints202604.0068.v1

$\epsilon = 0.008$ (time penalty), and $\kappa = 0.3$ (alignment bonus). Episodes terminate after 6000 steps. Together, these terms provide a dense and balanced learning signal that guides the agent toward socially aware, efficient navigation through rough terrain populated with dynamic pedestrians.

3.3.2. Environment Configuration

We conduct all experiments in the DUnE simulation testbed [18]. Utilizing the Gazebo Fortress simulation engine, we test in three different environments: Inspection, Construction, and Island (Figure 2). The robot used in testing is the FictionLab Leo Rover (0.985 m wheel-base, differential drive) [20]. Each scenario is initialized with a random goal position and 2–3 dynamic human actors that follow pre-scripted patrol waypoints within the same area as the random goal points.

(**a**) (**b**) (**c**)

Figure 2. The three environments used for our experiments: (a) Inspection, (b) Construction, and (c) Island. Inspection features sloped terrain with industrial obstacles, Construction has constrained navigation space with debris, and Island features uneven terrain with a few large obstacles. Inspection and Island contain three human actors each; Construction contains two.

3.3.3. Multi-Process Architecture Integration

A separate inference process takes camera images and applies a YOLOv11n-based pedestrian detector to each sampled frame. The resulting detections are converted into binary pedestrian masks, which are fused with the RGB inputs and passed to a spatiotemporal attention model. This model predicts a heatmap representing the likelihood of future human locations in the scene, leveraging both motion history and semantic context. Concurrently, the most recent RGB frame is processed by a monocular depth estimator (DepthAnythingV2) to obtain a normalized depth map. These three signals—grayscale+current pedestrian mask+predicted pedestrian attention heatmap, and depth image—are resized and fused into a $96 \times 96 \times 3$ tensor suitable for downstream reinforcement learning. The resulting observation is written to a third shared memory block (`rl_observation`) using an atomic update protocol that includes a timestamp and validity flag.

The overall architecture ensures isolation between perception and control, enabling asynchronous inference and action cycles. By relying on shared memory rather than message-passing, the system minimizes latency and avoids blocking between modules. Synchronization is enforced via timestamp-based freshness checks and atomic memory layout conventions. This integration strategy supports continuous, low-latency operation in real-time robotics environments, with clear modular boundaries between ROS-based frame acquisition, attention inference, and RL-based navigation.

3.3.4. RL Agent Image Observation Space Design

At each time t the agent receives a 96 pixel $\times$ 96 pixel $\times 3$ tensor

$$\mathbf{O}_t = \left[\mathbf{I}_t^{\text{gray}} \parallel \hat{H}_t \parallel \mathbf{D}_t\right], \tag{7}$$

where "$\parallel$" denotes channel-wise concatenation.

 doi:10.20944/preprints202604.0068.v1

- **Channel 1: Grayscale image with masks.** The current RGB frame is converted to luminance and normalised to $[-1, 1]$. Pixels inside the YOLO pedestrian bounding boxes are overwritten with the constant value -1 to emphasize occupied regions.
- **Channel 2: Human traversability heat-map** $\hat{H}_t \in [0, 1]^{96\times96}$ predicted by the attention model; values are logit-scaled to $(0, 1)$.
- **Channel 3: Depth map** $\mathbf{D}_t$ acquired from a simulated Intel Realsense at 30 Hz, clipped to 0–10 m and linearly mapped to $[-1, 1]$.

Before concatenation each channel is resized with bilinear interpolation and subjected to per-channel mean–variance normalisation computed over the training set. This fusion encodes both physical obstacles (depth), socially preferred regions (heat-map), and fine-grained appearance cues (grayscale) in a compact representation compatible with convolutional encoders. Figure 3 illustrates the information received by the RL agent.

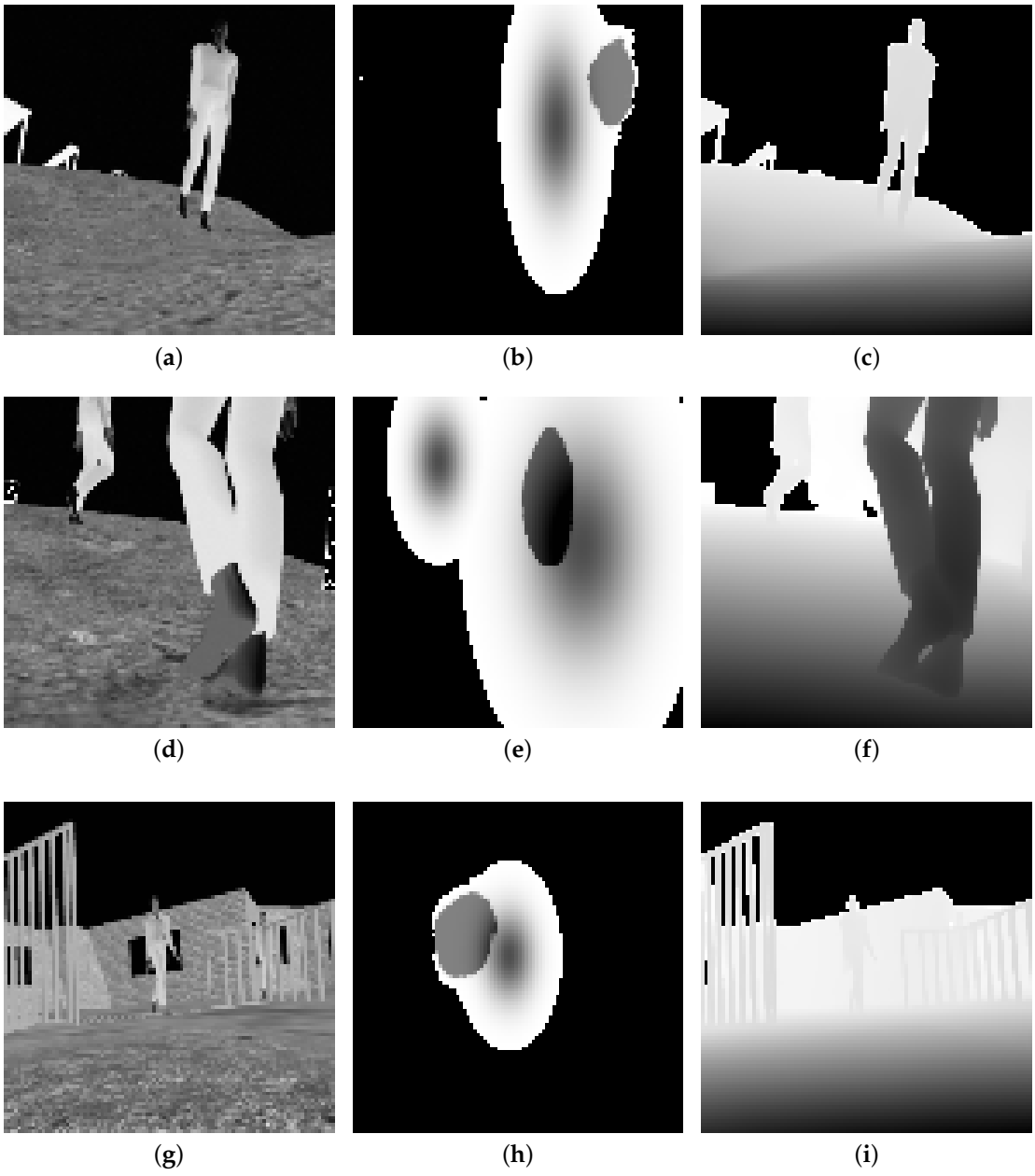

Figure 3. The three channels of the RL agent observation with grayscale (left), prediction heatmap with current and predicted future locations (middle), and depth (right). In the upper row, we can see the current and predicted positions in the heatmap. The middle row has two pedestrian detections.

 doi:10.20944/preprints202604.0068.v1

3.4. Perception Pipeline

At runtime the robot maintains a discrete set of K=5 view windows corresponding to fixed yaw offsets $\{-60°, -30°, 0°, +30°, +60°\}$ spanning approximately 160° of azimuthal coverage. The policy maintains a `vis_window` state variable indicating the currently selected window, and the action includes an incremental pan command (left / hold / right) that moves the window index by at most one view per step. Any view can therefore be reached within at most four decisions. Each raw fisheye frame is converted to a rectified, rectilinear window via a precomputed remap look-up table (LUT) bank containing one LUT per yaw bin. This rectification step is applied to the currently selected window before computing the downstream perceptual channels.

Given the rectified selected-view image, three perception modules run in parallel:

1. **YOLO inference** produces person detections, which are rendered as thick bounding-box borders that survive downsampling.
2. **Monocular depth inference** produces a geometry channel from the same selected view.
3. **Sobel edge magnitude** provides a lightweight structural signal that preserves obstacle boundaries and traversability-relevant texture at low resolution.

These outputs get combined (YOLO mask applied to the Sobel edge image) and concatenated with the grayscale image, to form a compact fused image $\mathbf{I}_t^{\text{fused}} \in \mathbb{R}^{96\times96\times3}$, which serves as the primary visual input to the RL agent.

The spatiotemporal attention mechanism requires a consistent view angle to maintain temporal coherence; it therefore always operates on the center view stream rather than the actively selected view. The attention network consumes a short RGB-plus-YOLO mask history from the center view and produces a dense occupancy heatmap indicating near-future human presence. This center-view heatmap is summarized into a compact density vector by concatenating column-wise and row-wise sums, producing a 192-dimensional feature:

$$\mathbf{h}_t \in \mathbb{R}^{192}.$$

This representation captures the spatial distribution of predicted human occupancy while remaining small enough to include directly in the RL observation dictionary.

At each step the RL process reads the latest observation from shared memory. The observation dictionary is structured as follows:

- **Fused image** $\mathbf{I}_t^{\text{fused}} \in \{0,\dots,255\}^{96\times96\times3}$: the stacked outputs of YOLO/Sobel, depth, and grayscale from the *selected* rectified view.
- **Heatmap-derived vector** $\mathbf{h}_t \in \mathbb{R}^{192}$: row and column sums of the *center-view* attention heatmap.
- **View token** `vis_window` $\in \{0,1,2,3,4\}$: the active window index.
- **IMU** $(\text{roll}, \text{pitch}, \text{yaw}) \in \mathbb{R}^3$: orientation from the onboard inertial measurement unit.
- **Target** $(\text{distance}, \text{relative azimuth}) \in \mathbb{R}^2$: goal-relative features for PointNav.
- **Velocities** $(\text{linear}, \text{angular}) \in \mathbb{R}^2$: current wheel velocities.

The agent acts in a flat discrete action space that is factored into a movement branch and a pan branch:

$$N_{\text{move}} \times N_{\text{pan}} = (5 \times 12) \times 3 = 180.$$

The movement branch enumerates five speed levels ($\{-0.5, 0.0, 0.2, 0.35, 0.7\}$ m/s) and twelve heading targets uniformly spaced in $[-\pi, \pi)$, while the pan branch provides three incremental options: pan left, hold, or pan right. A small penalty of -0.02 is applied to the reward function (5) when a non-zero pan is chosen to discourage random scanning and encouraging deliberate gaze shifts.

3.5. Curriculum Training

Because joint gaze-plus-navigation learning is brittle early in training, we employ a staged curriculum that gradually increases scene complexity while keeping the joint action coupling fixed

 doi:10.20944/preprints202604.0068.v1

throughout. As shown in Training begins in the **Inspection** environment with point-goal navigation over unstructured terrain and minimal obstacles, allowing the agent to acquire basic locomotion and goal-directed control. We then incrementally increase static obstacle density followed by progressive insertion of dynamic human actors (one, then two, then three). This staging encourages discovery of proactive gaze shifts that reduce surprise encounters while maintaining forward progress. The agent is trained in Inspection until convergence at approximately 1.6×10^7 environment steps.

After convergence in Inspection, the same agent (initialized from learned weights) is transferred to the **Island** environment, which emphasizes close-proximity human interactions with comparatively few static obstacles. This phase stresses social awareness and near-field proxemic behavior, refining gaze selection for early detection of pedestrians entering the robot's path. Training continues until convergence at approximately 3.6×10^7 steps.

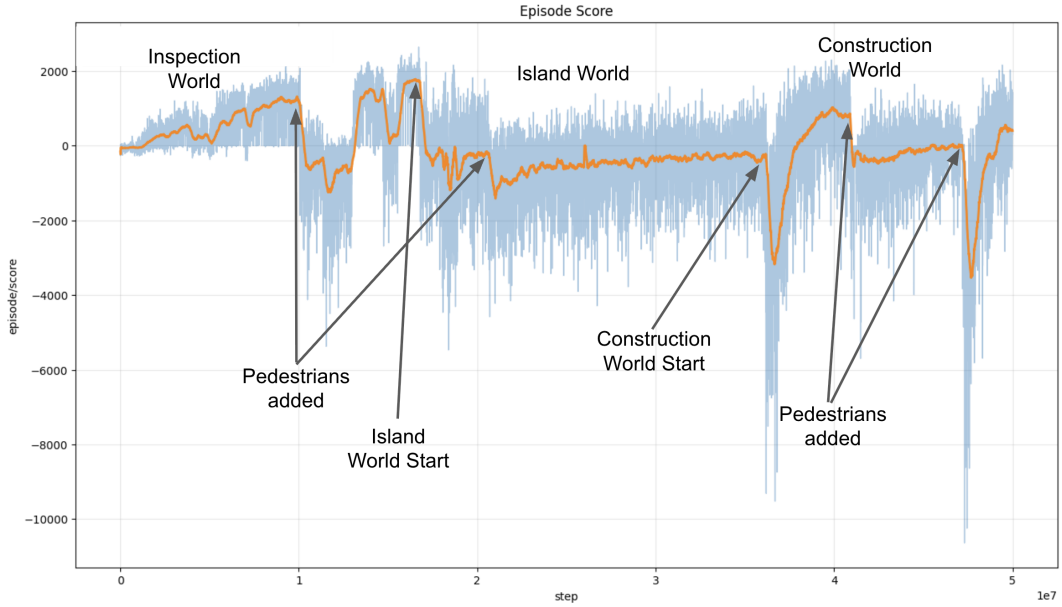

Figure 4. Episodic reward during curriculum training of the joint Active Vision + social-navigation RL agent policy. The agent is first trained in the **Inspection** world; task difficulty is increased by introducing dynamic pedestrians (and additional obstacles), producing a temporary reward drop followed by recovery as the coupled gaze–motion policy adapts. Training then transfers to the **Island** world (close-proximity interactions) and finally to the **Construction** world (dense clutter and constrained routes), each transition causing a sharp reward decrease and subsequent reconvergence. Blue shows per-episode returns and orange shows a smoothed trend.

Finally, the agent is transferred to the **Construction** environment, characterized by dense static clutter, tight corridors, two dynamic actors, and limited feasible routes. This stage couples constrained motion planning with human avoidance under severe occlusion and funneling effects. Training proceeds until convergence at approximately 5.0×10^7 steps.

3.6. Comparison Method: NAV2

Navigation2 (Nav2) is the standard open-source navigation framework for ROS 2, providing a modular, lifecycle-managed autonomy stack that integrates mapping/localization, planning, control, recovery behaviors, and behavior-tree task execution via replaceable plugins and standardized interfaces [21,22]. For global planning we use the Smac Planner family, which offers high-performance search-based planners (Cost-Aware A*, Hybrid-A*, and State Lattice) that enforce kinematic feasibility while leveraging cost fields to improve behavior in cluttered environments; Smac also exposes interpretable penalty terms (e.g., for non-straight motion, turn-direction changes, and reverse motion) and supports multiple motion models and goal-orientation modes, making it a strong, reproducible baseline within the ROS 2 ecosystem [23]. We compare against Nav2 as a geometric baseline because a 2-D LiDAR-centric stack provides inherently wide situational awareness that vision-only navigation

 doi:10.20944/preprints202604.0068.v1

often lacks: our configuration uses a full-azimuth planar scan (306 beams), enabling continuous obstacle detection, which is advantageous with moving pedestrians. We additionally evaluate a Nav2 w/ reverse variant that permits bidirectional motion. Under uniformly sampled PointNav goals, targets frequently fall behind the robot's current heading, and allowing reverse maneuvers avoids time-consuming in-place skid steer rotations in those cases. This stronger bidirectional variant gives Nav2 additional advantage for goal completion, providing a more rigorous baseline against which to evaluate AVSN's social navigation capabilities.

4. Results

To quantify changes in the robot's gaze while learning the social navigation policy, we plot the distribution of view windows. The window usage distribution plots shown in the upper row of Figure 5 and 6 summarize how often each discrete active-vision window is selected over an evaluation. Each bar corresponds to one window index and its height indicates the total number (or fraction) of timesteps for which that window was active. A highly peaked distribution implies the policy repeatedly favors a small subset of views (potentially indicating a persistent goal/obstacle bias or limited exploration of viewpoints), whereas a flatter distribution indicates more diverse sensing behavior with frequent switching across windows.

The bottom row of Figures 5 and 6 shows a histogram of the goal-alignment error, aggregated over all valid timesteps in the evaluation. Each bar counts the frequency of alignment errors within a fixed angular bin spanning 0° to 180° (36 bins total, i.e., ≈ 5° per bin), providing a compact view of how consistently the robot's active view is oriented toward the current navigation goal. Two reference thresholds are overlaid: 30° ("good" alignment) and 60° ("acceptable" alignment). Concentration of mass to the left of 30° indicates that the sensing direction is better aligned with the goal bearing, whereas a heavier tail beyond 60° indicates frequent misalignment and less goal-directed sensing.

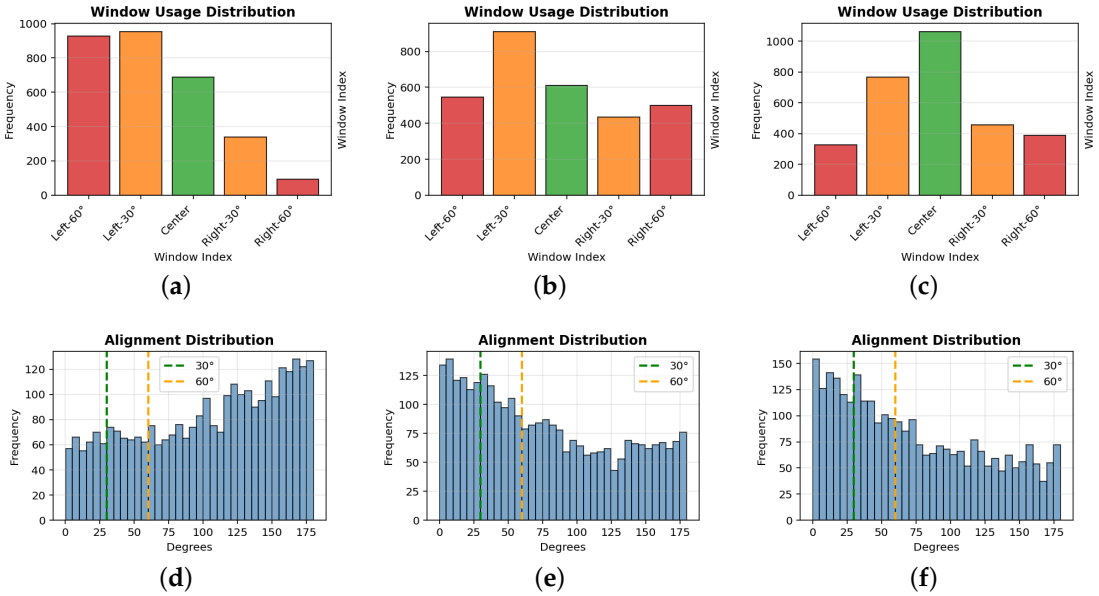

Figure 5. Initial alignment distribution as agent learns static obstacles. Top row: window selection histograms for (a) 1e6 steps, (b) 2e6 steps and (c) 3e6 steps. Bottom row: corresponding alignment distribution in degrees from looking towards the next goal which is a combination of robot orientation and view selection.

 doi:10.20944/preprints202604.0068.v1

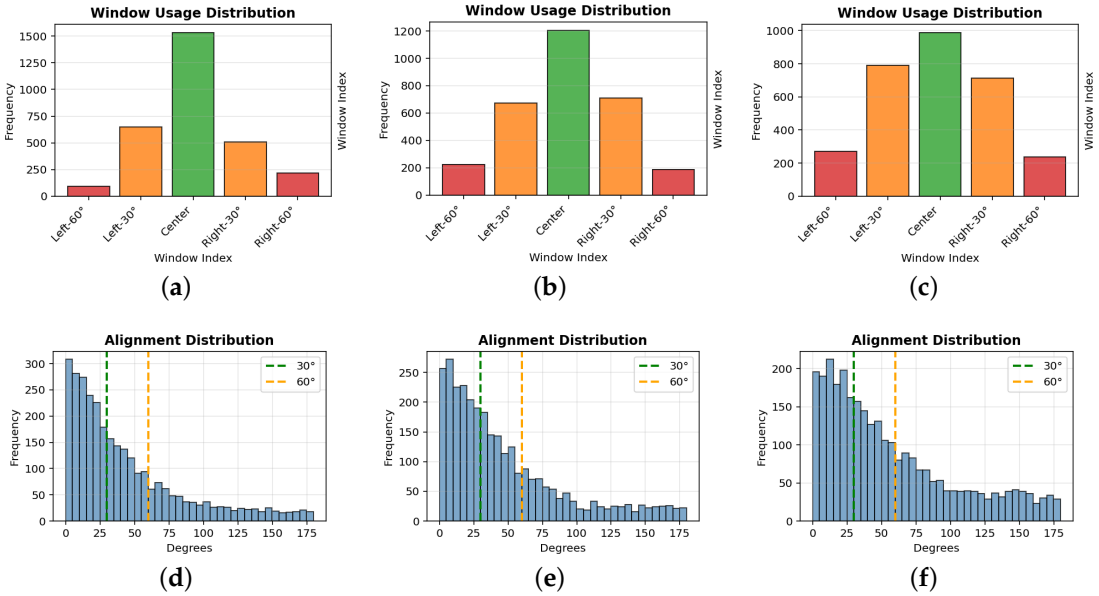

Figure 6. Window selection distribution and goal alignment starting with static obstacle convergence on the left (a) and progressing toward full social navigation convergence on the right (c). For static obstacles, the center window dominates the distribution while social navigation requires looking left and right to prevent social encounters.

Table 1 summarizes robot–human encounter statistics across the three Gazebo environments under six conditions: Attention, YOLO+Depth, No Social Nav, Nav2 (forward-only), Nav2 (with reverse enabled), and the proposed Active Vision variant. The YOLO+Depth condition provides human detections without attention-based future state prediction. Encounters are binned by minimum separation thresholds (0.5 m, 0.8 m, and 1.2 m), where smaller distances indicate higher proxemic risk, and task throughput is measured by total navigation goals completed. Active Vision yields the lowest critical close-proximity interaction counts ($d_{\min} < 0.5$ m) and the highest goals-per-encounter ratios across all environments. Measured by goals per $d_{\min} < 0.5$ m encounter, Active Vision outperforms Attention by 68%, 42%, and 145% in Inspection, Island, and Construction respectively, and outperforms YOLO+Depth by 115%, 122%, and 403%. These gains are achieved while maintaining comparable goal completion on Island and competitive throughput in Inspection and Construction, indicating that proactive gaze selection improves social safety without consistently sacrificing task performance. Active Vision reduces high-risk interactions relative to both classical navigation as well as our Attention method (Chapter 6) and ablations across all three unstructured environments.

To assess statistical differences between navigation conditions, we segmented each 90-minute evaluation into non-overlapping windows of $G = 10$ consecutive goals. For each window we computed risk seconds per goal, defined as the number of 1 Hz time steps during which the minimum robot–human distance fell below 0.5 m divided by G. Individual navigation episodes time out after 207 seconds, after which the robot and actors respawn at a randomly selected starting position, reducing spatial autocorrelation between episodes within a run. This goal-aligned windowing normalizes for differences in navigation speed across conditions, producing multiple observations per condition suitable for inferential testing.

For each environment we conducted six planned pairwise comparisons using the Mann–Whitney U test, a nonparametric rank-based test that does not assume normally distributed observations. Effect sizes are reported as Cohen's d with pooled standard deviation, and 95% confidence intervals for the difference in means use Welch–Satterthwaite degrees of freedom. To control the family-wise error rate across comparisons within each metric, we applied Holm–Bonferroni step-down correction ($\alpha = 0.05$).

 doi:10.20944/preprints202604.0068.v1

Table 1. Aggregate performance over 90-minute evaluations across three Gazebo environments (Inspection, Construction, Island) for six navigation conditions: Attention, YOLO+Depth, No Social Nav, Nav2 (forward-only), Nav2 (with reverse enabled), and Active Vision. Robot–human interactions are summarized as counts of time steps in which the minimum robot–human distance falls below 0.5 m, 0.8 m, and 1.2 m (smaller thresholds indicate higher proxemic risk). **Goals** reports the total number of navigation goals completed. **Goals per <0.5m** and **Goals per all <0.8m** normalize task throughput by close-proximity interactions (higher is better), where "all <0.8m" includes encounters with $d_{\min} < 0.5$ m.

Condition	<0.5m	<0.8m	<1.2m	Goals	Goals per <0.5m	Goals per all <0.8m
Construction						
Nav2 (forward)	92	65	124	278	3.02	1.77
Nav2 (w/reverse)	97	61	138	**320**	3.30	2.03
YOLO+Depth	14	17	77	142	10.14	4.58
No Social Nav	64	88	168	284	4.44	1.87
Attention (ours)	5	**11**	**54**	104	20.8	6.50
Active Vision (ours)	**3**	13	59	153	**51**	**9.56**
Island						
Nav2 (forward)	192	213	245	485	2.53	1.20
Nav2 (w/reverse)	236	217	261	**597**	2.53	1.32
YOLO+Depth	20	**40**	**137**	440	22	7.33
No Social Nav	214	208	247	531	2.48	1.26
Attention (ours)	13	46	139	448	34.46	7.59
Active Vision (ours)	**11**	45	186	537	**48.82**	**9.59**
Inspection						
Nav2 (forward)	206	250	239	196	0.95	0.43
Nav2 (w/reverse)	196	211	283	197	1.01	0.48
YOLO+Depth	77	84	123	361	4.69	2.24
No Social Nav	108	108	129	**439**	4.06	2.03
Attention (ours)	54	74	**121**	324	6	2.53
Active Vision (ours)	**29**	**39**	126	293	**10.10**	**4.31**

As shown in Table 2, across all three environments, Active Vision consistently reduced proxemic risk, measured as *risk seconds per goal* (time steps with $d_{\min} < 0.5$ m normalized by goals). In **Inspection**, Active Vision achieved significantly lower risk than both the No Social Nav baseline ($\Delta = -0.1825$, $p_{\text{corr}} < 0.001$) and Nav2 (w/reverse) ($\Delta = -0.4456$, $p_{\text{corr}} < 0.001$), and also improved over Attention ($\Delta = -0.0722$, $p_{\text{corr}} = 0.01$) (Table 1). In **Island**, where close-proximity interactions dominate, Active Vision produced a large and significant reduction in risk relative to No Social Nav ($\Delta = -0.4308$, $p_{\text{corr}} < 0.001$) and Nav2 (w/reverse) ($\Delta = -0.3919$, $p_{\text{corr}} < 0.001$), while remaining statistically indistinguishable from Attention and YOLO+Depth. In the most constrained **Construction** world, Active Vision again significantly reduced risk compared to No Social Nav ($\Delta = -0.2407$, $p_{\text{corr}} < 0.001$) and Nav2 (w/reverse) ($\Delta = -0.3316$, $p_{\text{corr}} < 0.001$), but did not differ from Attention. Overall, the primary social-navigation outcome is that both attention-based policies and Active Vision substantially decrease time spent within 0.5 m of pedestrians compared to baselines lacking social reasoning, with the strongest improvements observed against Nav2 and No Social Nav across environments.

 doi:10.20944/preprints202604.0068.v1

Table 2. Statistical comparisons for Construction, Island and Inspection environments using goal-normalized windows ($G = 10$ goals per window). Δ: difference in means (A−B). d: Cohen's d. p_{corr}: Holm–Bonferroni corrected. Risk secs per goal: seconds with $d_{\min} < 0.5$ m per goal achieved (lower = safer). * $p < .05$, ** $p < .01$, *** $p < .001$.

Social Encounters per Goal ($d_{\min} < 0.5$ m)	Δ	d	p_{corr}	
Construction World				
Attention vs YOLO+Depth	-0.0808	-0.68	0.26	
Attention vs No Social Nav	-0.2107	-1.14	0.002	**
Attention vs Nav2 (w/reverse)	-0.3016	-1.10	0.008	**
Active Vision vs Attention	-0.0300	-0.65	0.26	
Active Vision vs No Social Nav	-0.2407	-1.39	<0.001	***
Active Vision vs Nav2 (w/reverse)	-0.3316	-1.28	<0.001	***
Island World				
Attention vs YOLO+Depth	-0.0216	-0.28	0.71	
Attention vs No Social Nav	-0.4262	-2.57	<0.001	***
Attention vs Nav2 (w/reverse)	-0.3874	-1.90	<0.001	***
Active Vision vs Attention	-0.0045	-0.08	0.71	
Active Vision vs No Social Nav	-0.4308	-2.72	<0.001	***
Active Vision vs Nav2 (w/reverse)	-0.3919	-2.00	<0.001	***
Inspection World				
Attention vs YOLO+Depth	-0.0684	-0.60	0.01	*
Attention vs No Social Nav	-0.1103	-0.66	0.01	*
Attention vs Nav2 (w/reverse)	-0.3734	-1.72	<0.001	***
Active Vision vs Attention	-0.0722	-0.63	0.01	*
Active Vision vs No Social Nav	-0.1825	-1.11	<0.001	***
Active Vision vs Nav2 (w/reverse)	-0.4456	-2.04	<0.001	***

5. Conclusions

This article demonstrates the value of active vision for social navigation. In our implementation, viewpoint selection is a jointly learned decision variable alongside locomotion. The key contributions are: (1) a discrete gaze mechanism over five yaw-offset rectified views spanning 160°, integrated into a flat 180-action space; (2) a dual-stream perception architecture that decouples temporally consistent center-view social prediction from actively controlled selected-view scene understanding; (3) a compact observation interface combining a $96\times96\times3$ fused image, a 192-D heatmap density vector, a view token, and proprioceptive features; and (4) a three-stage curriculum (Inspection → Island → Construction) that stabilizes the inherently difficult joint gaze-plus-navigation credit assignment.

Experimental evaluation across three Gazebo environments demonstrated that AVSN consistently achieves the lowest critical encounter counts ($d_{\min} < 0.5$ m) and the highest goals-per-encounter ratios among all tested conditions. Statistical testing confirmed that Active Vision significantly reduces proxemic risk relative to both the No Social Nav baseline and Nav2 (w/reverse) in all three environments ($p_{\text{corr}} < 0.001$), while in Inspection it also improved significantly over the Attention-only condition ($p_{\text{corr}} = 0.01$). These results confirm that proactive gaze control substantially improves social compliance in unstructured environments without consistently sacrificing task throughput.

Author Contributions: Conceptualization, J.V. and G.S.; software, J.V.; investigation, J.V.; writing—original draft preparation, J.V.; writing—review and editing, G.S.; visualization, J.V.; supervision, G.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

 doi:10.20944/preprints202604.0068.v1

Institutional Review Board Statement: Not applicable

Data Availability Statement: Source code available at https://github.com/jackvice/attention, https://github.com/jackvice/dreamerJ https://github.com/jackvice/RoboTerrain

Acknowledgments: During the preparation of this study, the authors used ChatGPT5 and Gemini 3 for the purposes of grammar correction and paragraph structuring. The authors have reviewed and edited the output and take full responsibility for the content of this publication.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

CNN	Convolutional Neural Network
FOV	Field of View
LSTM	Long Short Term Memory
MPC	Model Predictive Control
RL	Reinforcement Learning
YOLO	You Only Look Once

References

1. Shang, J.; Ryoo, M.S. Active vision reinforcement learning under limited visual observability. *Advances in Neural Information Processing Systems* **2023**, *36*, 10316–10338.
2. Dass, S.; Hu, J.; Abbatematteo, B.; Stone, P.; Martín-Martín, R. Learning to look: Seeking information for decision making via policy factorization. *arXiv preprint arXiv:2410.18964* **2024**.
3. Wang, G.; Li, H.; Zhang, S.; Guo, D.; Liu, Y.; Liu, H. Observe then act: Asynchronous active vision-action model for robotic manipulation. *IEEE Robotics and Automation Letters* **2025**.
4. Grimes, M.K.; Modayil, J.V.; Mirowski, P.W.; Rao, D.; Hadsell, R. Learning to look by self-prediction. *Transactions on Machine Learning Research* **2023**.
5. Hafner, D.; Pasukonis, J.; Ba, J.; Lillicrap, T. Mastering diverse control tasks through world models. *Nature* **2025**, pp. 1–7.
6. Rios-Martinez, J.; Spalanzani, A.; Laugier, C. From proxemics theory to socially-aware navigation: A survey. *International Journal of Social Robotics* **2015**, *7*, 137–153.
7. Dautenhahn, K. Socially intelligent robots: Dimensions of human-robot interaction. *Philosophical Transactions of the Royal Society B: Biological Sciences* **2007**, *362*, 679–704.
8. Hall, E.T. *The Hidden Dimension*; Anchor Books: New York, NY, USA, 1966.
9. Thrun, S.; Fox, D.; Burgard, W. The dynamic window approach to collision avoidance. *IEEE Robotics and Automation Magazine* **1997**, *4*, 23–33.
10. Alonso-Mora, J.; Breitenmoser, A.; Rufli, M.; Beardsley, P.; Siegwart, R. Optimal reciprocal collision avoidance for multiple non-holonomic robots. In *Distributed Autonomous Robotic Systems: The 10th International Symposium*; Springer, 2013; pp. 203–216.
11. Charalampous, K.; Kostavelis, I.; Gasteratos, A. Robot navigation in large-scale social maps: An action recognition approach. *Expert Systems with Applications* **2016**, *66*, 261–273.
12. Vega, A.; Cintas, R.; Manso, L.J.; Bustos, P.; Núñez, P. Socially-accepted path planning for robot navigation based on social interaction spaces. In Proceedings of the Iberian Robotics Conference. Springer, 2019, pp. 644–655.
13. Tai, L.; Zhang, J.; Liu, M.; Burgard, W. Socially compliant navigation through raw depth inputs with generative adversarial imitation learning. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Brisbane, Australia, 2018; pp. 1111–1117.
14. Xie, Z.; Dames, P. DRL-VO: Learning to navigate through crowded dynamic scenes using velocity obstacles. *IEEE Transactions on Robotics* **2023**, *39*, 2700–2719.
15. Golchoubian, M.; Ghafurian, M.; Dautenhahn, K.; Azad, N.L. Uncertainty-aware DRL for autonomous vehicle crowd navigation in shared space. *IEEE Transactions on Intelligent Vehicles* **2024**.
16. Wang, Y.; Xie, Y.; Xu, D.; Shi, J.; Fang, S.; Gui, W. Heuristic dense reward shaping for learning-based map-free navigation of industrial automatic mobile robots. *ISA Transactions* **2025**, *156*, 579–596.

 doi:10.20944/preprints202604.0068.v1

17. Bae, J.; Kim, J.; Yun, J.; Kang, C.; Choi, J.; Kim, C.; Lee, J.; Choi, J.; Choi, J. SiT Dataset: Socially Interactive Pedestrian Trajectory Dataset for Social Navigation Robots. In Proceedings of the 37th Conference on Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks Track, New Orleans, LA, USA, 2023.
18. Vice, J.M.; Sukthankar, G. DUnE: A Versatile Dynamic Unstructured Environment for Off-Road Navigation. *Robotics* **2025**, *14*, 35.
19. Yang, L.; Kang, B.; Huang, Z.; Zhao, Z.; Xu, X.; Feng, J.; Zhao, H. Depth Anything V2, 2024, [arXiv:cs.CV/2406.09414].
20. Ali, N. Computer vision-guided autonomous grasping system using Leo Rover with robotic arm. Master's thesis, University of South-Eastern Norway, 2024.
21. Macenski, S.; Martín, F.; White, R.; Ginés Clavero, J. The Marathon 2: A Navigation System. In Proceedings of the 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2020.
22. Macenski, S.; Moore, T.; Lu, D.; Merzlyakov, A.; Ferguson, M. From the desks of ROS maintainers: A survey of modern and capable mobile robotics algorithms in the robot operating system 2. *Robotics and Autonomous Systems* **2023**.
23. Macenski, S.; Booker, M.; Wallace, J. Open-Source, Cost-Aware Kinematically Feasible Planning for Mobile and Surface Robotics. *Arxiv* **2024**.