A. Appendix of GRIDTOPIX: Training Embodied Agents with Minimal Supervision

In this appendix we include:

A.1 Details of reward structure for {PointNav, FurnMove, 3 *vs.* 1 with Keeper }×{shaped, terminal} (Tab. A.1).
A.2 An alternate terminal reward structure for FurnMove by simply dropping a component from $r_t^{\perp\!\!\!\perp}$ (Tab. A.2).
A.3 Additional quantitative metrics for FurnMove introduced in [29] (terminal rewards – Tab. A.3 and shaped rewards – Tab. A.4).
A.4 Edits to allow deep net models to encode gridworld observation (*vs.* visual observations).
A.5 Teacher forcing probability and IL → RL transition for GRIDTOPIX and GRIDTOPIX→DIRECTPIX routines (Tab. A.5).
A.6 Experimental results for Visual Predator-Prey task in OpenAI multi-particle environment [46, 49]. This serves as a testbed wherein *reward shaping is not enough* (Fig. A.1 and Tab. A.6).
A.7 Qualitative visualizations of policy learned by DirectPix agent in 3 *vs.* 1 with Keeper (Fig. A.2 and Fig. A.3).

A.1. Reward Structures

In Tab. A.1, we list all the reward components for shaped and terminal rewards for the three tasks considered in this work. For a fair comparison, the shaped rewards are kept identical to prior work in PointNav [58], FurnMove [29], and 3 *vs.* 1 with Keeper [38]. For terminal rewards, as per the definitions in Sec. 3.1, there exists no r_t^{progress} component (hence, '✘'). The results for these reward structures have been reported in the main paper (Tab. 1 and Tab. 2).

Common across all tasks, $\mathcal{I}[\text{success}]$ denotes the indicator function conditioned on success of the episode. 'Step penalty' is a small negative reward to encourages completion of the episode in fewer steps. In PointNav, the progress reward is based on the shortest-path geodesic distance to the goal from the current location of the agent (d_t^{geodesic}). In FurnMove, particularly in [29], the authors use a variant of the PointNav progress reward. Particularly, Manhattan distances are used ($d_t^{\text{Man.}}$) and a positive reward is received only if the agents get the furniture item closer to the goal compared to the minimum distance in previous steps (note the use of '$\max\min$'). For 3 *vs.* 1 with Keeper, the progress reward is called the 'checkpoint reward'. It is received if the agents are able to move the ball to a zone closer to the goal.

Task	**Reward structure**	$r_t^{\textbf{success}}$	$r_t^{\textbf{progress}}$	$r_t^{\perp\!\!\!\perp}$
PointNav	Shaped	$10 \cdot \mathcal{I}[\text{success}]$	$d_{t-1}^{\text{geodesic}} - d_t^{\text{geodesic}}$	Step penalty (-0.01)
PointNav	Terminal	$10 \cdot (1 - 0.9 \cdot \frac{t}{T}) \cdot \mathcal{I}[\text{success}]$	✘	0
FurnMove	Shaped	$1 \cdot \mathcal{I}[\text{success}]$	$\max(\min_{k=0,\ldots,t-1} d_k^{\text{Man.}} - d_t^{\text{Man.}}, 0)$	Failed action penalty (-0.02) Failed coordination (-0.1) Step penalty (-0.01)
FurnMove	Terminal	$1 \cdot \mathcal{I}[\text{success}]$	✘	Failed action penalty (-0.02) Failed coordination (-0.1) Step penalty (-0.01)
3 *vs.* 1 with Keeper	Shaped	$1 \cdot \mathcal{I}[\text{success}]$	Checkpoint reward (0.1)	0
3 *vs.* 1 with Keeper	Terminal	$1 \cdot \mathcal{I}[\text{success}]$	✘	0

Table A.1. **Reward structures.** For each task (PointNav, FurnMove, and 3 *vs.* 1 with Keeper), we list the three components of shaped and terminal reward structures. This includes a positive reward conditioned on success r_t^{success}, a goal-dependent progress reward r_t^{progress}, and a goal-independent reward $r_t^{\perp\!\!\!\perp}$. Terminal rewards, that we focus on in this work, do not include progress reward (see Sec. 3.1 for definitions). Hence, progress reward is marked with a '✘' for terminal settings.

A.2. Alternate Terminal Reward Structures

As shown in Tab. A.1, the terminal reward structure for FurnMove is equal to the shaped reward structure [29] except that it does not include the progress reward component r_t^{progress}. All results in the main paper (FurnMove column of Tab. 1 and Tab. 2) and Tab. A.3 and Tab. A.4 correspond to this terminal reward setting.

We, additionally, study an alternative terminal reward formulation where we drop the failed action penalty from the reward structure, *i.e.*, only step penalty and failed coordination constitute $r_t^{\perp\!\!\!\perp}$. We found the metrics to improve with this reward structure, with only 250k episode of training. These results are summarized in Tab. A.2. By further dropping the failed coordination

Training Routine	**MD-SPL**	**Success**
DirectPix	0.1	2.5
GRIDTOPIX	6.8	44.5
GRIDTOPIX → DirectPix	3.4	18.6
Gridworld expert (*upper bound*)	*17.0*	*66.8*

Table A.2. Quantitative results for terminal rewards without failed action penalty (FurnMove).

penalty, the training became sample-inefficient. This is coherent with the findings of Jain *et al*. [29]. Particularly, given the *tightly-coupled* nature of the agents in collaboratively moving furniture, the failed coordination reward is critical for efficient learning.

A.3. Additional FurnMove Metrics

In the main paper, we report the primary metrics of % successful episodes (Success) and a Manhattan distance based SPL (MD-SPL). For a fair comparison, we report three additional metrics that were included in [29]. This includes number of actions taken per agent (*Ep Length*), probability of uncoordinated actions (*Invalid Prob Mass*), and meters from goal at the episode's end (*Final Distance*). Tab. A.3 and Tab. A.4 supplement the results reported in Tab. 1 and Tab. 2, respectively.

Training Routine	MD-SPL ↑	Success ↑	Ep Length ↓	Invalid Prob Mass ↓	Final Distance ↓
DirectPix	0.0	0.8	249.3	0.150	3.42
GridToPix	**4.0**	**24.6**	**210.7**	**0.110**	**2.848**
GridToPix → DirectPix	3.1	14.5	224.2	0.116	3.215
Gridworld expert (*upper bound*)	*19.2*	*56*	*139.0*	*0.077*	*1.943*

Table A.3. **Additional quantitative rewards (terminal reward structure).** This table supplements results reported in Tab. 1. In addition to metrics of MD-SPL and success rate, we include other relevant metrics. Vertical arrows, *i.e.*, ↑ and ↓ denotes whether larger or smaller metric values are preferred, respectively.

Training Routine	MD-SPL ↑	Success ↑	Ep Length ↓	Invalid Prob Mass ↓	Final Distance ↓
DirectPix	11.2	58.4	155.7	0.311	1.154
GridToPix	9.7	62.0	154.6	0.264	1.17
GridToPix → DirectPix	**15.3**	**68.6**	**133.6**	**0.213**	**0.826**
Gridworld expert (*upper bound*)	*22.2*	*76.3*	*109.7*	*0.275*	*0.722*

Table A.4. **Additional quantitative rewards (shaped reward structure).** This table supplements results reported in Tab. 2.

A.4. Gridworld Encoder and Implementation Details

The models utilized for visual and gridworld agents are alike. Particularly, we make minimal edits to adapt the observation encoder to be able to process gridworld tensors instead of RGB tensors. This is briefly summarized below.

PointNav. For the PointNav, the visual encoder transforms a $(3, 256, 256)$ RGB tensor into a feature of length 512 via three convolutional blocks and a linear layer. The grid encoder transforms a $(1, 100, 100)$ top-down tensor into a feature of the same length as the visual counterpart (512) via four convolutional blocks and a linear layer. For both visual and gridworld agents, the policy is represented via a GRU of hidden size 512 followed by linear layers to serve as actor and critic heads.

FurnMove. Similarly as for PointNav, for FurnMove the visual encoder transforms a $(3, 84, 84)$ RGB tensor into a feature of length 512 via five convolutional blocks (no linear layers). The grid encoder transforms a $(9, 15, 15)$ top-down tensor into a feature of length 512 via four convolutional blocks (for consistency, no linear layers). For both visual and gridworld agents in FurnMove task, the policy is represented via a LSTM of hidden size 512 followed by linear layers to serve as actor and critic heads. Note that we use the (best-performing) mixture-of-marginals actor head introduced in [29] for all FurnMove experiments.

3 *vs*. 1 with Keeper. For the 3 *vs*. 1 with Keeper task, the $(3, 1280, 960)$ RGB tensor is scaled to a $(1, 96, 72)$ gray-scale image. The visual encoder transforms this $(1, 96, 72)$ tensor into a feature of length 512 via three convolution layers and a linear layer. The gridworld model observes the state as a vector of length 572 and transforms it to a feature of length 64 using a three linear layers. For both visual and gridworld agents, linear layers on top of the extracted feature serve as actor and critic heads.

A.5. Training Routines

In the main paper we compare our GridToPix and GridToPix→DirectPix routines with DirectPix (Sec. 5.1). Here, we include different exploration policies for imitation learning. We also include additional details of teacher forcing [76, 7] and IL → RL transition (GridToPix→DirectPix).

Exploratory policies μ. Recall, from Sec. 3.3, the GridToPix loss is

$$\mathcal{L}_{\text{GridToPix}} = \mathbb{E}[\mathbb{E}_{a\sim\pi^{\mathcal{G}}(\cdot|O_\mu,H_\mu)}[-\log \pi^{\mathcal{V}}(a \mid O_\mu, H_\mu)]]. \tag{5}$$

The choice of exploratory policy μ leads to three variants, each widely adopted in IL tasks:
• Student forcing (SF): This is an *on-policy* method, *i.e.*, the target policy $\pi^{\mathcal{V}}$ to be learnt is the exploration policy μ.
• Teacher forcing (TF): In this case we use $\mu = \pi^{\mathcal{G}}$, *i.e.*, the visual agent takes the expert's actions. This helps the visual agent frequently observe states closer to the goal, which it would only see late in training if following SF. The downside of TF: the visual agent $\mathcal{V}$ observes only states which meaningfully lead to the goal. Hence, TF is susceptible to *covariate shift*, *i.e.*, the visual agent $\mathcal{V}$ exhibits low resilience to recover if it ventures 'off-track.'
• Annealed teacher forcing or DAgger (DA): Agents take actions by combining SF and TF via $a_t^{\mathcal{V}} = (1-\beta)a_t^{\text{SF}} + \beta a_t^{\text{TF}}$ where $\beta \sim \text{Bernoulli}(p)$. A decay of p from 1 to 0 is adopted during training to transition smoothly from TF to SF. After such annealing, the visual agent's policy $\pi^{\mathcal{V}}$ is generally trained with SF until convergence.

Teacher forcing probability. Details of teacher forcing for our methods are summarized in Tab. A.5. We employ annealed teacher forcing (see Sec. 3.3) for a part of the training budget. We anneal (decay) the teacher forcing probability linearly for PointNav, FurnMove, and exponentially for the 3 *vs.* 1 with Keeper.

The GRIDTOPIX routine is purely IL and is denoted in Tab. A.5 with an IL arrow (⟷) spanning from 0% to 100%. In contrast, GRIDTOPIX→DIRECTPIX includes a warm-start with IL followed by reward-based learning. Hence, in Tab. A.5, we have a shorter IL arrow followed by an RL arrow (⟷) along with the algorithm used to maximize rewards.

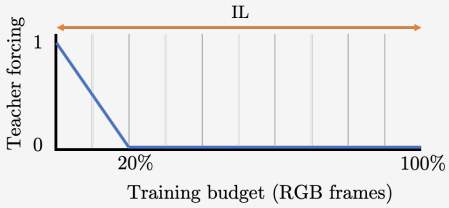
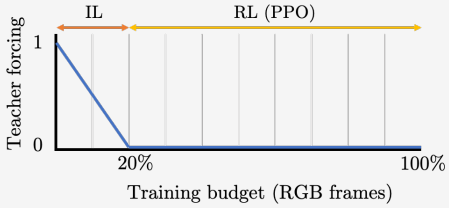
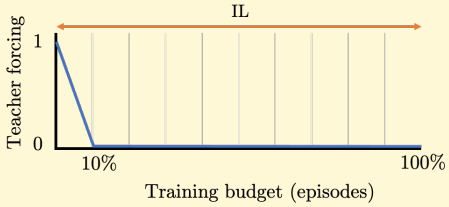
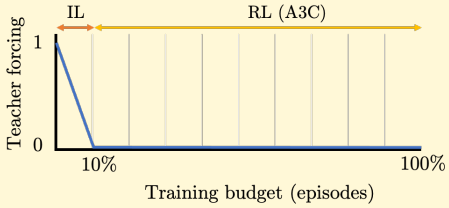
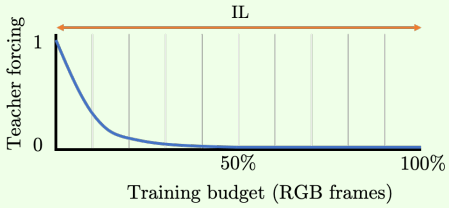
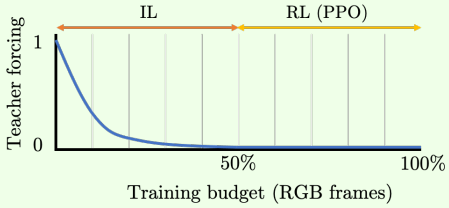
Task	GRIDTOPIX	GRIDTOPIX→DIRECTPIX
PointNav		
FurnMove		
3 *vs.* 1 with Keeper		

Table A.5. **Training routines.** The figures show how teacher forcing probability varies over training and the transition from IL to RL.

A.6. Visual Predator-Prey – Task Where Reward Shaping Is Not Enough

Predator-prey is a multi-agent task defined within the OpenAI multiple-particle environments (MPE) [46]. It entails controlling a team of predators while a competing team of prey is controlled by the game engine. Specifically, n predators work together to chase a team of $n/3$ prey that move faster than the predators. The objective is to optimize rewards by controlling the policies of the predators. Prior work [46, 43, 33] assumes an agent observes a 1D vector summarizing the positions and velocities of all agents in a neighborhood. For consistency, we consider this 1D vector to be our *gridworld* observation. In addition, akin to [41], we define an analogous *visual* setting where agents only process a top-down map in pixel space.

Note, for visual tasks like PointNav, FurnMove, and 3 *vs.* 1 with Keeper, we created (or leveraged existing) gridworlds. As standard visual tasks are mostly navigational, reward shaping is typically tractable. Hence, to build a testbed where reward shaping isn't tractable, we are effectively creating the visual world for the complex, multi-agent predator-prey task.

Below, we include experimental details and corresponding results. Note, despite basic reward shaping, DirectPix agents

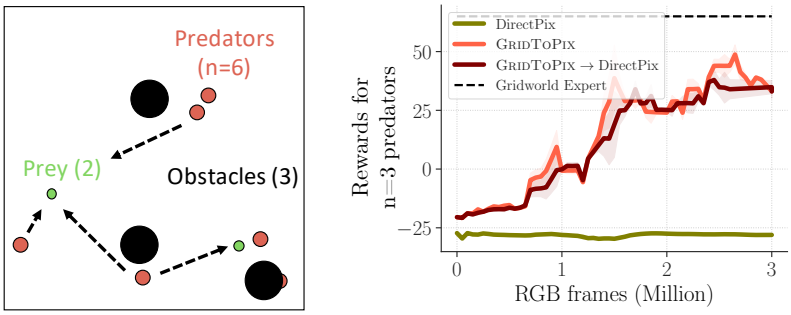


Figure A.1. Visual Prey-Predator: task setup and learning curve.

Method	**Reward** ($n = 3$)	
	@10%	*@100%*
DirectPix	-27.6	-28.0
GridToPix	-16.9	48.7
GridToPix→DirectPix	-17.1	37.9
Grid expert *(upper bound)*	65.0	

Table A.6. Visual *predator-prey* for num. predators $= n = 3$.

cannot learn whereas GridToPix comes close to the upper bound of gridworld experts.

Gridworld observation. The predator observes its location and velocity, the relative location of the neighboring landmarks and fellow predators, and the relative location and velocity of the three preys.

Shaped rewards. Due to the complexity of the task, it's intractable to *perfectly* shape the reward. However, a positive reward for bumping into a prey and a negative reward based on the distance to the prey is provided.

Model architecture. We use a standard CNN model [48]. Our model has four hidden layers. The first three layers are convolutional layers with $32, 64$, and 64 filters. All convolutional layers have filters of size 4×4 and stride 2. The fourth layer is a fully connected layer with 256 hidden units. Following the last hidden layer are linear layers that predict the value and actor policy.

Evaluation. Agents are evaluated using the average rewards obtained in test episodes. We train the agents for 3M environment steps. Learning curves are reported on test episodes.

Results. We experiment with number of predators $= n = \{3, 6\}$ visual predator-prey tasks. The learning curves for $n = 3$ and setup for $n = 6$ are illustrated in Fig. A.1. The average rewards for DirectPix, GridToPix, GridToPix→DirectPix, and gridworld experts are included in Tab. A.6. Despite basic reward shaping, joint optimization of perception and planning leads to a DirectPix policy demonstrating no meaningful behaviour. With the help of self-supervision via the gridworld expert, GridToPix and GridToPix→DirectPix perform significantly better with 48.7 and 37.9 average rewards over the training budget of 3M steps. Results on the $n = 6$ setting show a similar trend: DirectPix and GridToPix obtain average rewards of -70.2 and 237.1 (the gridworld expert earns 281.1).

A.7. Qualitative Results of DirectPix Trained with Terminal Rewards

As we report in Sec. 5, DirectPix doesn't learn a meaningful policy in any of the tasks when given only terminal rewards. Closer inspection reveals that the DirectPix agent for the PointNav task learns a degenerate probability distribution with almost all probability mass allocated to the 'Stop' action. Similarly, many of the strategies learned by DirectPix agents for 3 *vs*. 1 with Keeper are also myopic. Particularly, the agents cannot effectively pass to each other, pushing the ball outside the field lines (see Fig. A.2). Another common failure mode is shooting at the goal from too far off (see Fig. A.3).

Figure A.2. **Failure modes – Misdirected passes.** Two episodes (top strip and bottom strip) that highlight a common failure mode of DirectPix training with terminal rewards. Particularly, player 1 attempts to pass the ball to player 2 but misdirects it to hit outside the field lines. For readability, the relevant players are marked as P1 and P2. Also, ball is highlighted using a yellow arrow.

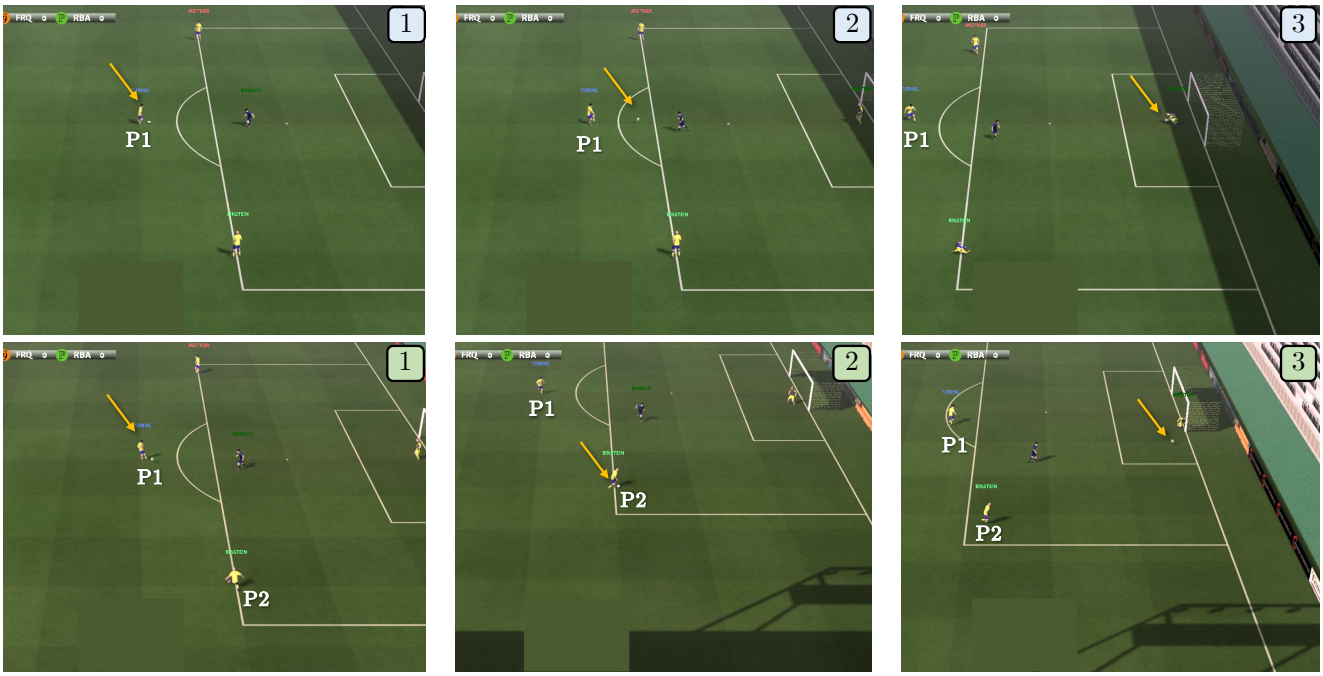


Figure A.3. **Failure modes – Long distance shooting.** Two episodes (top strip and bottom strip) that highlight a common failure mode of DirectPix training with terminal rewards. Particularly, in the top strip, player 1 attempts to score by shooting too ambitiously from the starting point. In the bottom strip, player 2 does the same, after receiving the ball from player 1. In both episodes the goal keeper can easily intercept the ball.