TacSIm: A Dataset and Benchmark for Football Tactical Style Imitation

Supplementary Material

1. Reconstruction Results and Visualizations

Broadcast trajectories inevitably contain missing fragments due to occlusion, shot changes and detection failures. Given partially observed sequences $X^{\text{obs}} \in \mathbb{R}^{N\times T\times 2}$ and an observation mask M, the goal is to reconstruct the complete states X^{full} for all players (and ball) on the field while preserving physical plausibility and team-level coherence. All trajectory–completion models are trained Alfheim Dataset, with an optional augmentation from the Google Research Football simulator to increase coverage of rare motion patterns. The Alfheim Dataset comprises two-dimensional player coordinates captured by sensors worn by athletes.

We conducted experiments using the Recurrent Conditional Variational Autoencoder (RC-VAE) method on the Alfheim Dataset and the Alfheim Dataset combined with gfootball. The Alfheim Dataset comprises 80,000 training sequences and 10,000 test sequences, each annotated with 2D coordinates (x, y) over 50 time steps. The integration with gfootball adds 50,000 training sequences and 5,000 test sequences, incorporating cooperative multi-agent dynamics from soccer scenarios with similar temporal resolution. For both datasets, we introduced masking sequences with 40 to 50 missing values per trajectory.

Table 1 presents the performance of the RC-VAE method on the Alfheim Dataset and the Alfheim-gfootball Dataset. Metrics include the averaged L2 loss for prediction (P-L2) and imputation (I-L2). On the Alfheim Dataset, the RC-VAE achieves a P-L2 of 8.92 and an I-L2 of 5.59, reflecting its ability to predict future trajectories and impute missing values based on observation windows. On the Alfheim-gfootball Dataset, the P-L2 is 9.45, and the I-L2 is 6.12, indicating a slight increase in error due to the added complexity of cooperative dynamics.

The slight increase in P-L2 and I-L2 on the Alfheim-gfootball Dataset suggests that cooperative multi-agent dynamics introduce additional complexity, yet the model remains robust. The schematic diagram of the reconstruction site is shown in Figure 2. These results indicate that the off-the-shelf RC-VAE is well-suited for our reconstruction task, enhancing the credibility of reconstructed data within the dataset.

2. Baseline Model Training Parameters

Table 2 presents the detailed parameter configurations for all baseline imitation-learning models, consolidated into a single stacked layout (left column: model name; right column: Architecture/Optimizer/LR/Settings) for readability.

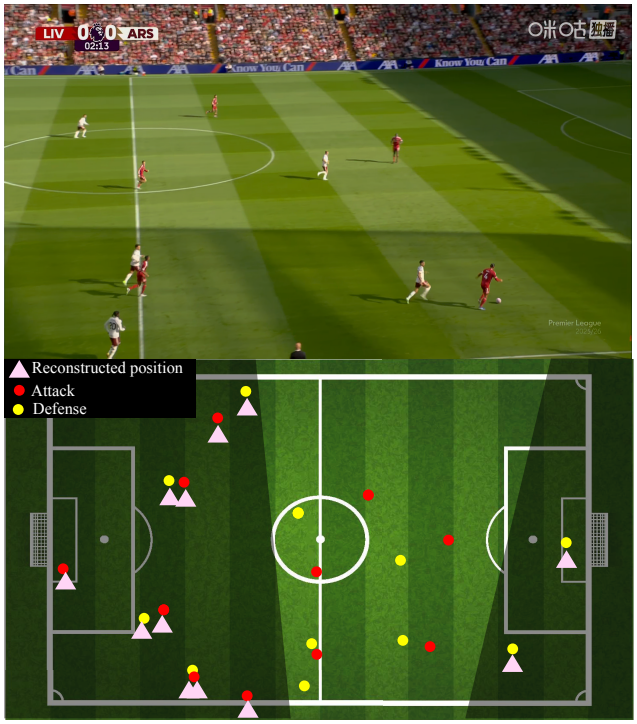

Figure 2. Player position reconstruction.

Table 1. Performance of RC-VAE method on Alfheim Dataset and Alfheim-gfootball Dataset.

Dataset	P-L2 ↓	I-L2 ↓
Alfheim Dataset	8.92	5.59
Alfheim-gfootball Dataset	9.45	6.12
Ground Truth (GT)	0.00	0.00

Note: P-L2 and I-L2 denote averaged L2 loss for prediction and imputation, respectively. ↓ indicates lower is better.

3. Multi-resolution Grid-Based Spatial Visualization of Ball Trajectories

As shown in Figure 3, side-by-side comparison of results for the same scene across three prediction durations (3.0/5.0/10.0s) and six grid resolutions. Each subgraph overlays actual and generated trajectories, with color blocks indicating grid occupancy to facilitate observation of spatial coverage and directional consistency changes over time.

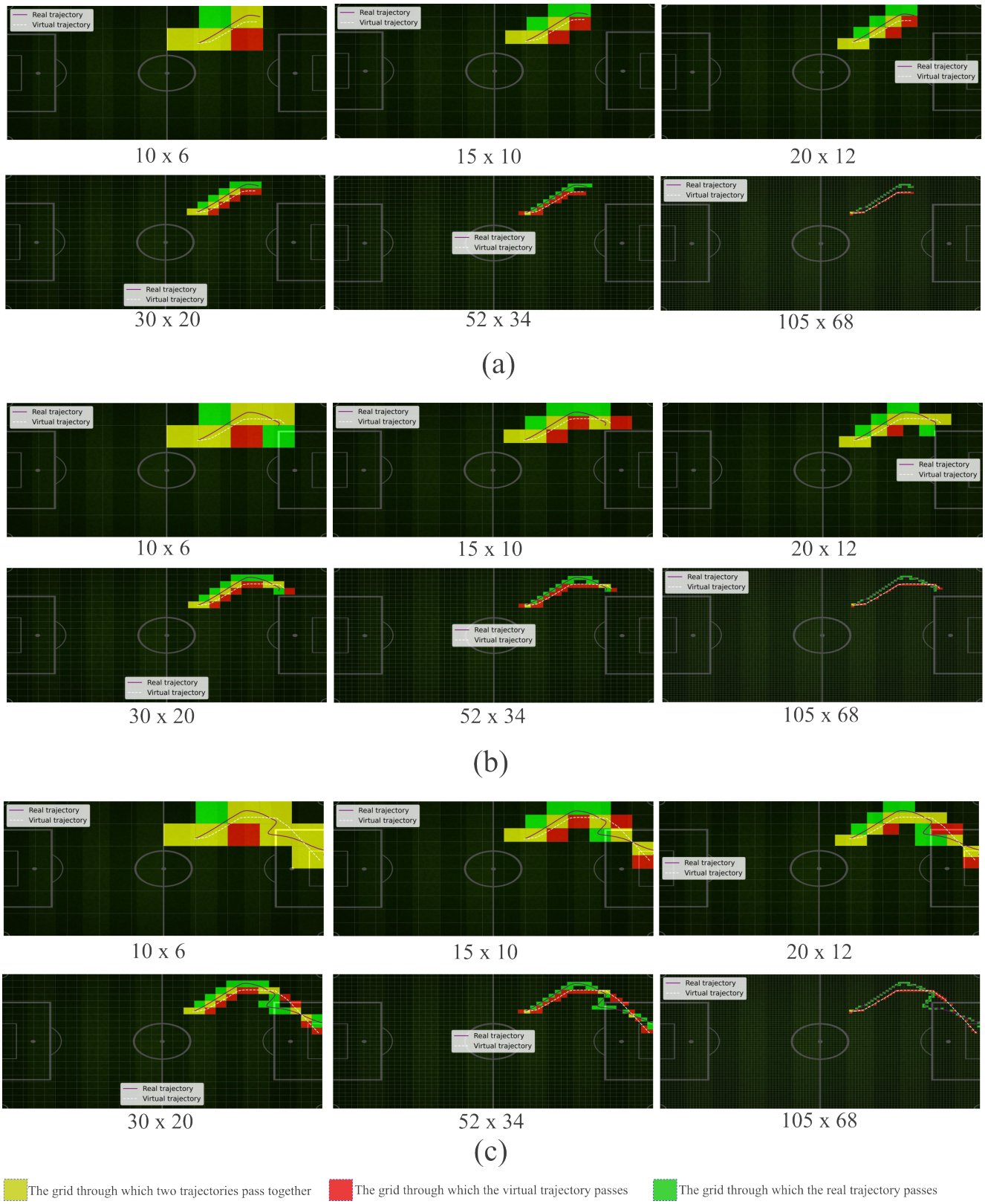

Figure 3. Schematic of the football trajectory imitation. Where (a) is the imitation trajectory of the displayed 3s, (b) is the imitation trajectory of the displayed 5s, and (c) is the imitation trajectory of the displayed 10s. The figure shows the evaluation for each of the six grid sizes.

Table 2. Parameter configurations for all baselines.

BC	**Architecture** Per-agent MLP (shared weights), 4 layers, hidden dim 256; input: 50-frame context features; output: 25-step 2D displacements per player; layers: Linear $\rightarrow$ ReLU $\rightarrow$ LayerNorm (final layer linear). **Optimizer** Adam **LR** 1×10^{-4} (cosine decay, 2-epoch warmup) **Settings** Activation: ReLU; Loss: position L2 + velocity L2 (w=0.1); weight decay 1×10^{-4}; dropout 0.1; grad clip 1.0; batch size 32. Training: variable context $L \in \{1, 5, 10, 25, 50\}$; short closed-loop rollout (10 steps); observation 50 / prediction 25; windows slide within possession/phase boundaries.
CMIL	**Architecture** Latent role assignment (HMM-style, K=11) with Hungarian matching; per-agent MLP policies (3 layers, hidden 256); joint-state encoder (teammates/opponents + ball). **Optimizer** Adam **LR** 1×10^{-4} **Settings** Loss: imitation (L2) + coordination regularizer; role prior temperature 1.0; update role assignment once per epoch; entropy reg 1×10^{-3}; dropout 0.1; weight decay 1×10^{-4}. Training matches BC: variable context, short closed-loop (10 steps), obs 50 / pred 25, batch 32.
IRL (AIRL-style)	**Architecture** **Policy**: per-agent MLP (256) predicting 25-step displacements; Discriminator $D(s, a, s')$: MLP (256), reward $r = \log D - \log(1 - D)$; optional time encoding to aggregate the 50-frame context. **Optimizer** Adam (policy & discriminator) **LR** 1×10^{-4} for both; alternating updates 1:1 **Settings** PPO: clip $\epsilon = 0.2$, $\gamma = 0.99$, GAE $\lambda = 0.95$, entropy coef 5×10^{-3}, value coef 0.5; rollout length 25; reward normalization (EMA); optional gradient penalty; dropout 0.1; weight decay 1×10^{-4}. Same variable context and short closed-loop as above; batch 32.
CoDAIL	**Architecture** Decentralized per-agent MLP(2$\times$256) with opponent-modeling head; multiple GAIL-style discriminators (per-agent or per-group) to distinguish real vs. generated team trajectories; stabilization via coordination priors. **Optimizer** Adam (policy & discriminators) **LR** 1×10^{-4} (cosine decay) **Settings** Loss: adversarial (discriminator) + imitation (supervised component); entropy coef 5×10^{-3}; optional gradient penalty & reward normalization; PPO/BC mixing as in implementation; dropout 0.1; weight decay 1×10^{-4}. Training protocol same as above.
DRAIL	**Architecture** **Diffusion classifier** as discriminator: UNet-MLP head (hidden 256), diffusion steps T=1000, cosine noise schedule; Policy: per-agent MLP (256). Reward computed as $r = \log D - \log(1 - D)$. **Optimizer** Adam (policy & diffusion classifier) **LR** 1×10^{-4} for both (linear or cosine decay) **Settings** Loss: adversarial + diffusion (MSE to noise target); PPO: clip ϵ=0.2, γ=0.99, GAE λ=0.95, value coef 0.5, entropy coef 0–10^{-3}; reward normalization / EMA stabilization; dropout 0.1; weight decay 1×10^{-4}. Same training protocol (variable context, short closed-loop, obs 50 / pred 25, batch 32).

4. In-camera position reconstruction error verification

We validate localization via an image-plane proportion test(figure 4). We only use frames showing the full pitch and manually mark the key field lines (touchlines and halfway line) to avoid errors from automatic line extraction. We apply the same procedure to GRF-rendered broadcasts and compare the resulting ratio distributions using standard distribution distances, W1 (Wasserstein-1 in [0,1])(Table 3). This indicates our mapped coordinates show no obvious systematic bias and are reliable.

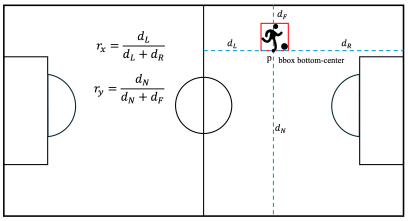

Figure 4. Image-plane proportion test schematic diagram.

Table 3. Alignment of player-to-field-line distance ratios.

Ratio	Real (mean±std)	GRF (mean±std)	W1↓
r_x (touchlines)	0.49 ± 0.18	0.50 ± 0.17	0.035
r_y (goallines)	0.53 ± 0.21	0.52 ± 0.20	0.045

We quantify this effect with a stratified evaluation: on frames where TVCalib succeeds (main broadcast view), we split players by image vertical position (bottom half vs top half, as a surrogate for camera distance) and report pitch-coordinate errors per group(Table 4). We find the far group is noisier, but the gap is small relative to our grid resolution and does not change benchmark conclusions.

Table 4. Granularity vs distance-to-camera (broadcast view).

Group (proxy)	Mean L2 (m)↓	P90 (m)↓
Bottom half (near)	2.34	6.85
Top half (far)	3.18	7.91

Our dataset is in continuous pitch coordinates; grid resolution is only a deterministic evaluation discretization for computing occupancy statistics at different tactical scales, not a component that affects data generation. We include multiple grids to cover the granularity–robustness trade-off (coarser grids are less sensitive to noise; finer grids give more detail).

5. Off-camera position reconstruction error verification

Synthetic broadcast validation (GRF). We use the invisible player position prediction method RC-VAE to predict visible positions and evaluate the prediction accuracy of RC-VAE. This shows RC-VAE reconstructs hidden-player positions with stable meter-level errors on GRF ground truth.

Table 5. Position accuracy of hidden players in GRF. **Unit: meter**

Mask	Average P-L2↓	Average I-L2↓
Broadcast	8.14	6.27

Acceptable accuracy. We add random offset to the RC-VAE predictions and compare against the average 3-second score of the CoDAIL method under the 240-grid protocol. This indicates that small offsets on imputed player positions have limited impact on imitation performance (Table 6).

Table 6. Imitation under localization random offset.**Unit: meter**.

Random offset	P-L2↓	I-L2↓	Imitation score↑
0	6.82	4.91	43.22
5	11.82	9.91	40.69
10	16.82	14.91	35.12

Baseline comparison. Trajectron++ [2] and Social-GAN [1] are forecasting models, so we evaluate them with prediction error (P-L2) and also report ADE/FDE (standard in the original papers and applicable to RC-VAE) (Table 7). These results show that RC-VAE outperforms Social-GAN and Trajectron++, achieving lower prediction errors and better reconstruction of hidden states.

Table 7. Identical masked reconstruction. **Unit: meter**.

Method	Dataset	Mask	P-L2↓	I-L2↓	ADE↓	FDE↓
Social-GAN [1]	Alfheim	Broadcast	10.60	–	10.60	13.61
Trajectron++ [2]	Alfheim	Broadcast	9.85	–	9.85	12.24
RC-VAE (ours)	Alfheim	Broadcast	**8.92**	**5.59**	**8.92**	**11.45**
Social-GAN [1]	GRF Data	Broadcast	8.97	–	8.97	12.84
Trajectron++ [2]	GRF Data	Broadcast	8.20	–	8.20	10.63
RC-VAE (ours)	GRF Data	Broadcast	**7.02**	**5.40**	**7.02**	**10.39**

References

[1] Agrim Gupta, Justin Johnson, Li Fei-Fei, Silvio Savarese, and Alexandre Alahi. Social gan: Socially acceptable trajectories with generative adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2255–2264, 2018. 4

[2] Tim Salzmann, Boris Ivanovic, Punarjay Chakravarty, and Marco Pavone. Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data. In *European conference on computer vision*, pages 683–700. Springer, 2020. 4