KW-DYAN: A Recurrent Dynamics-Based Network For Video Prediction

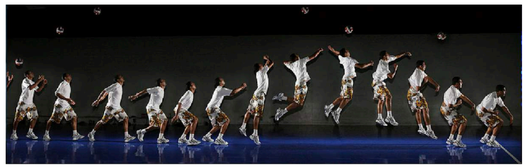
Wenqian Liu, Armand Comas, Yuexi Zhang, Octavia Camps, Mario Sznaier
Department of Electrical & Computer Engineering, Northeastern University, Boston, USA

Robust Systems Lab

Northeastern University

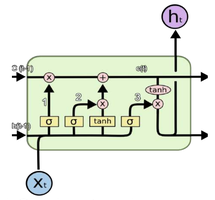
Motivation

DYNAMICS:

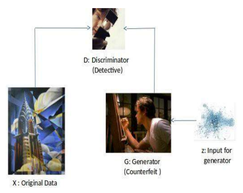


Provide powerful cues to understand scenes and anticipate the future:

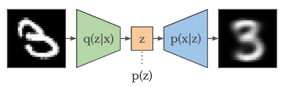
Video Frame Prediction Approaches



LSTM networks



GAN networks

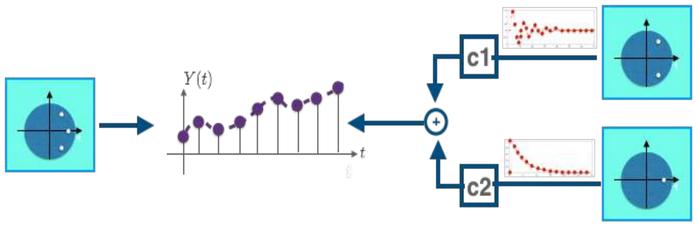


VAE networks

- Large number of parameters ✗
- Difficult to train ✗
- Blurry images ✗

Dynamics-based Representation

Model temporal signals as the output of Linear Time Invariant (LTI) systems:

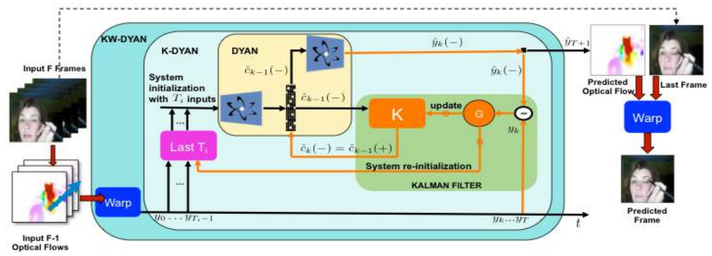


$$Y(z) = \sum_{i=1}^{n} \frac{c_i z}{z - p_i} \quad \Leftrightarrow \quad y(k) = \sum_{i=1}^{n} c_i {p_i}^k$$

- Can be represented by n pairs (c_i, p_i) coefficient-pole
- The number of poles n is the memory of the system
- "Simpler" systems have fewer poles

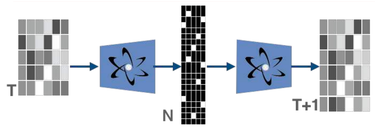
This work was supported in part by NSF grants IIS-1814631, ECCS-1404163, CMMI-1638234 and CNS-1646121; AFOSR grant FA9550-15-1-0392; and the Alert DHS Center of Excellence under Award Number 2013-ST-061-ED0001.

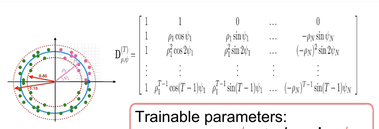
KW-DYAN: A recurrent network to capture dynamics



- Small number of parameters ✓
- Easy to train ✓
- Sharp images ✓
- Accurate timing ✓
- Recurrent: can process arbitrary long sequences ✓

Uses a Dynamics-based Sparse Autoencoder





Trainable parameters:
$p_i = \rho_i \cos\psi_i + j\rho_i \sin\psi_i$

Encoder: Uses a *structured* dictionary to find the least number of poles and coefficients to encode the optical flow inputs:

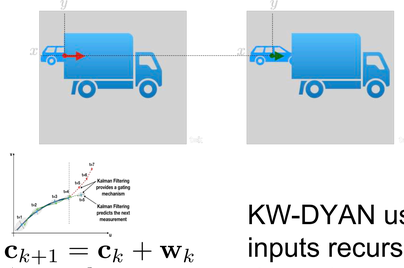
$$\mathbf{c}^* = \arg\min_{\mathbf{c}} \frac{1}{2}\|\mathbf{y}_{1:T} - \mathbf{D}_{\rho,\psi}^{(T)}\mathbf{c}\|^2 + \lambda\|\mathbf{c}\|_1$$

Fidelity — Sparsity

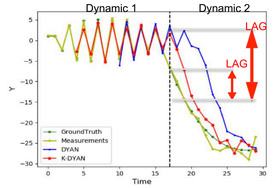
Decoder: Extends the dictionary to decode the sparse code and predict future optical flows:

$$\mathbf{y}_{1:T+1} = \mathbf{D}_{\rho,\psi}^{(T+1)}\mathbf{c}^*$$

Detects Changes in Dynamics

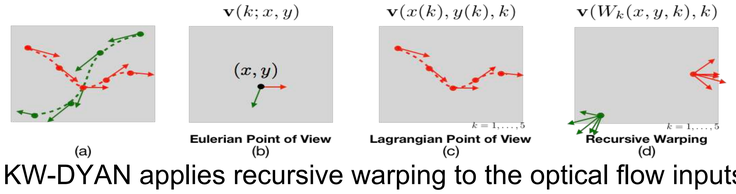


$$\mathbf{c}_{k+1} = \mathbf{c}_k + \mathbf{w}_k$$
$$\hat{y}_{k+1} = \mathbf{d}_{k+1}\mathbf{c_k} + v_{k+1}$$



KW-DYAN uses a Kalman filter to process its inputs recursively, detect changes in dynamics, and reduce prediction lagging.

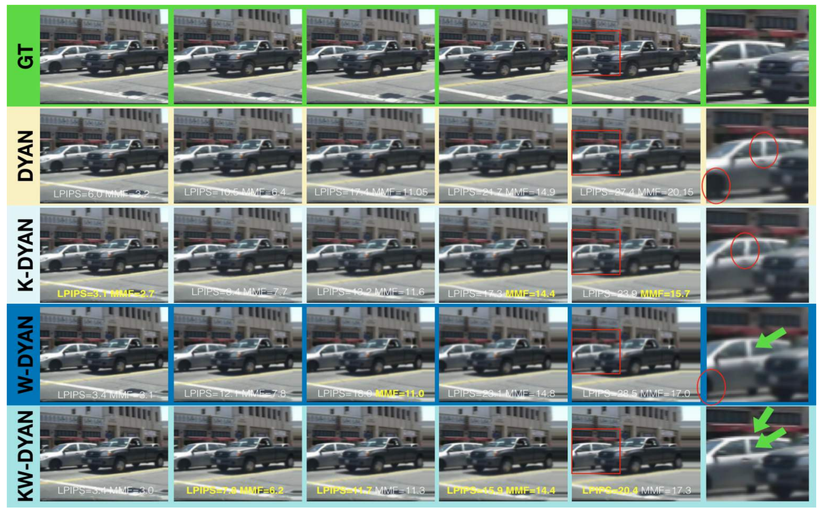
Reduces the Number of Changes in Dynamics

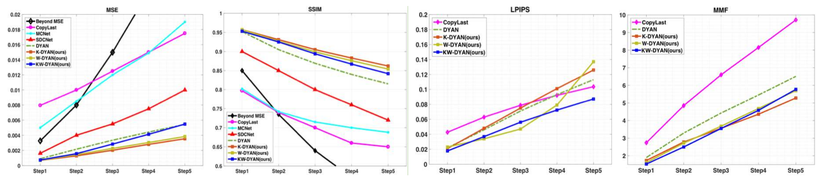


KW-DYAN applies recursive warping to the optical flow inputs to combine Lagrangian and Eulerian point of views and reduce the number of changes in dynamics that cause lagging.

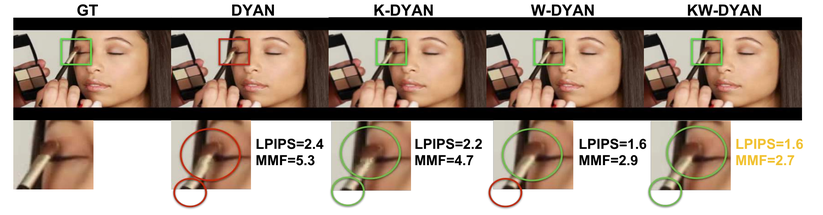
Experiments

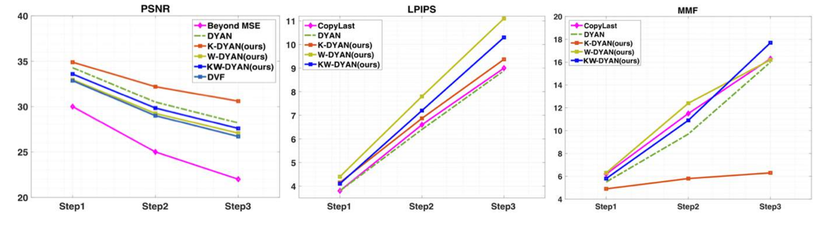
CALTECH-Car Mounted camera video dataset (10 frames input)



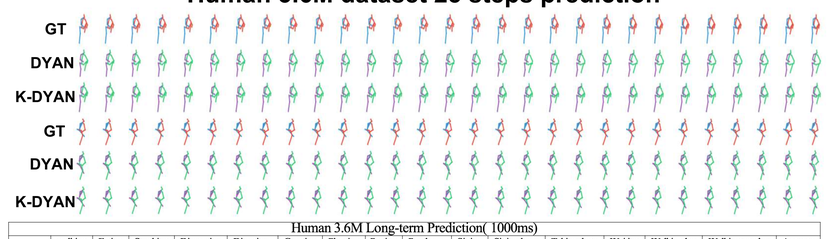


UCF101: Human action video dataset (4 frames input)





Human 3.6M dataset 25 steps prediction



Human 3.6M Long-term Prediction(1000ms)

	walking	Eating	Smoking	Discussion	Directions	Greeting	Phoning	Posing	Purchases	Sitting	Sittingdown	Takingphoto	Waiting	Walkingdog	Walkingtogether	Average
RRNN	1.14	1.34	1.83	1.79	1.59	2.03	1.89	2.56	2.30	2.14	2.72	1.51	2.34	1.86	1.42	1.90
CSS	0.92	**0.92**	**1.62**	1.86	1.45	1.72	1.81	2.65	2.52	1.67	2.06	1.40	2.5	1.92	1.28	1.77
TP-RNN	**0.77**	1.14	1.66	1.74	1.38	1.81	**1.68**	2.47	2.28	1.74	1.93	1.35	2.46	1.98	**1.28**	1.71
DYAN	1.21	1.34	1.86	1.72	1.44	1.84	1.83	2.40	2.33	1.95	2.30	1.44	2.35	1.84	1.56	1.83
KDYAN	1.17	1.34	1.68	**1.67**	**1.37**	**1.72**	1.76	**2.38**	**2.27**	**1.64**	**1.90**	**1.31**	**2.32**	**1.82**	1.45	**1.70**