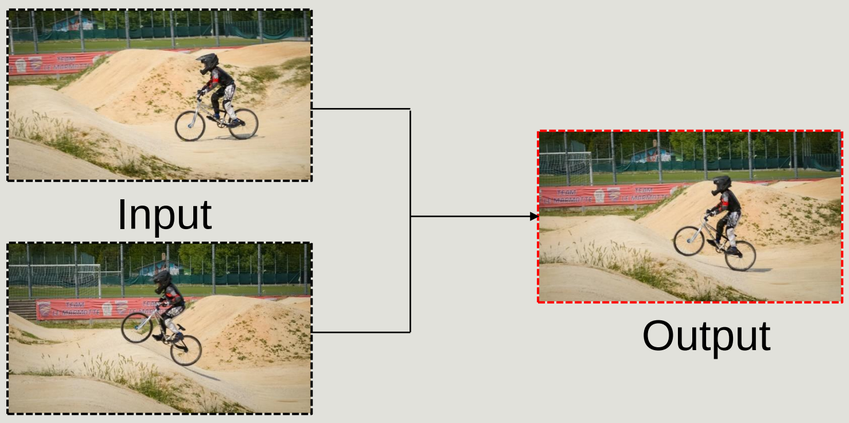
Residual Learning of Video Frame Interpolation Using Convolutional LSTM

Keito Suzuki and Ikehara Masaaki (Keio University)

Introduction:

Video frame interpolation is the task of generating an intermediate frame between 2 consecutive frames.

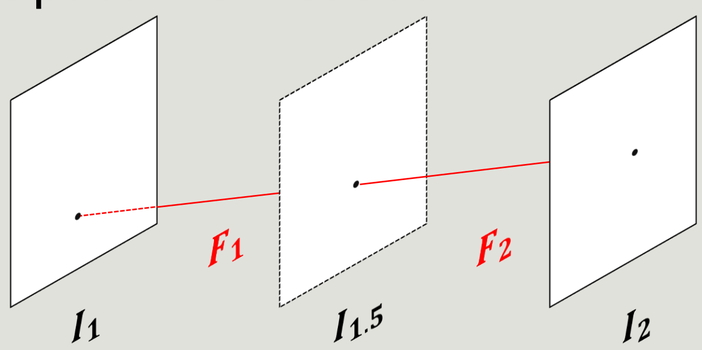


Goal: Propose a frame interpolation method that doesn't use optical flow or kernel estimation.

Conventional Methods:

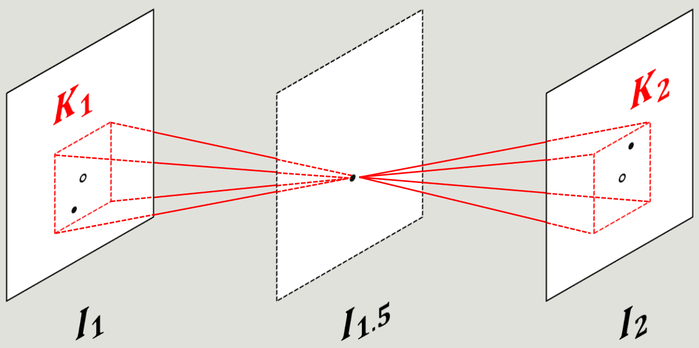
2 common ways of frame interpolation:

1) Optical flow based



- Difficult to obtain ground truth flow data
- Not as accurate on natural images

2) Kernel based



- Can consume a lot of memory
- Cannot handle motion larger than the kernel size

→Intermediate steps can limit the performance of frame interpolation

Proposed Method:

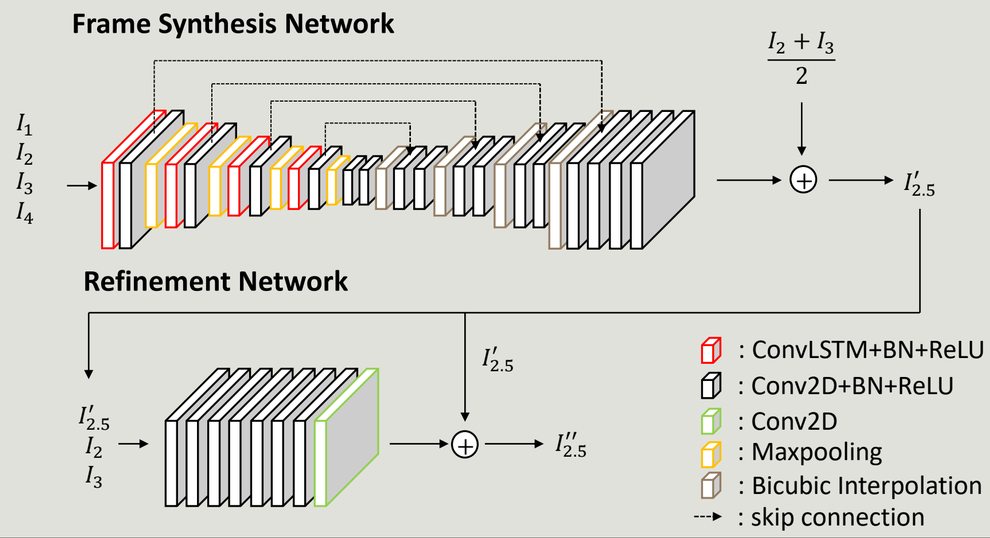
Directly generate the intermediate frame:

- Use residual learning

I_1 I_2 Difference

- Convolutional LSTM for feature extraction

Network structure:



Results:

Comparison on Middlebury Other, Vimeo90K, DAVIS 2017 Datasets

	Middlebury Other		Vimeo90K		DAVIS 2017	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
SepConv	35.25	0.960	33.38	0.939	27.73	0.864
CyclicGen	32.45	0.921	32.08	0.916	28.44	0.880
DAIN	36.42	0.968	34.57	0.952	28.23	0.873
Ours	35.48	0.958	35.41	0.953	28.67	0.881

Effectiveness of our model

Video	Baseline		+ConvLSTM		+Residual		+ConvLSTM +Residual		Complete Model	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
DogDance	29.25	0.869	29.91	0.884	33.26	0.940	33.32	0.939	33.41	0.938
Minicooper	25.72	0.874	26.52	0.892	30.69	0.963	30.73	0.963	31.52	0.969
RubberWhale	35.70	0.947	37.13	0.951	43.32	0.986	43.52	0.986	43.96	0.987
Walking	32.30	0.944	33.18	0.945	36.30	0.962	36.30	0.962	36.52	0.965
Beanbags	26.89	0.874	28.61	0.889	29.94	0.921	29.94	0.921	30.30	0.927
Hydrangea	32.16	0.888	32.92	0.918	35.40	0.951	36.12	0.953	36.71	0.971
Grove2	27.69	0.812	28.24	0.843	33.32	0.948	33.75	0.954	34.91	0.965
Grove3	25.24	0.792	26.15	0.835	28.61	0.908	28.71	0.909	29.40	0.923
Urban2	30.22	0.817	31.53	0.862	31.81	0.880	32.05	0.883	32.27	0.886
Urban3	30.70	0.824	32.93	0.883	35.37	0.924	35.22	0.922	36.02	0.934
Average	29.59	0.864	30.71	0.890	33.80	0.938	33.97	0.939	34.50	0.947