

# Super SloMo:High Quality Estimation of Multiple Intermediate Frames for Video Interpolation

s5 知的學習論研究室 中島弘樹

# 目次

- 1. 背景
- 2. 既存手法
- 3. 提案手法
- 4. 評価
- 5. まとめ

# 目次

- 1. 背景
- 2. 既存手法
- 3. 提案手法
- 4. 評価
- 5. まとめ

# 背景

かっこいいー！



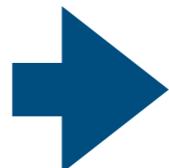
かっこいいけど、  
この技どうなってるんだ…??



ゆっくり再生したけどカクカクでわからん…



# 背景



1secで再生されるフレーム数が減って、  
動画内の動きがカクカクになる

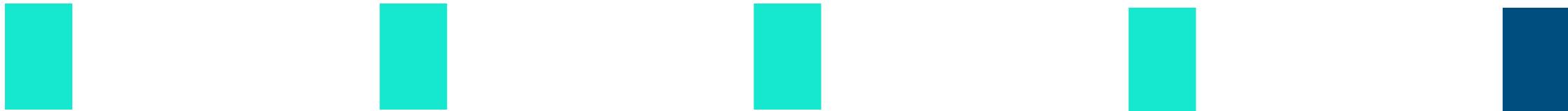
Original

20fps



Slow motion

4fps



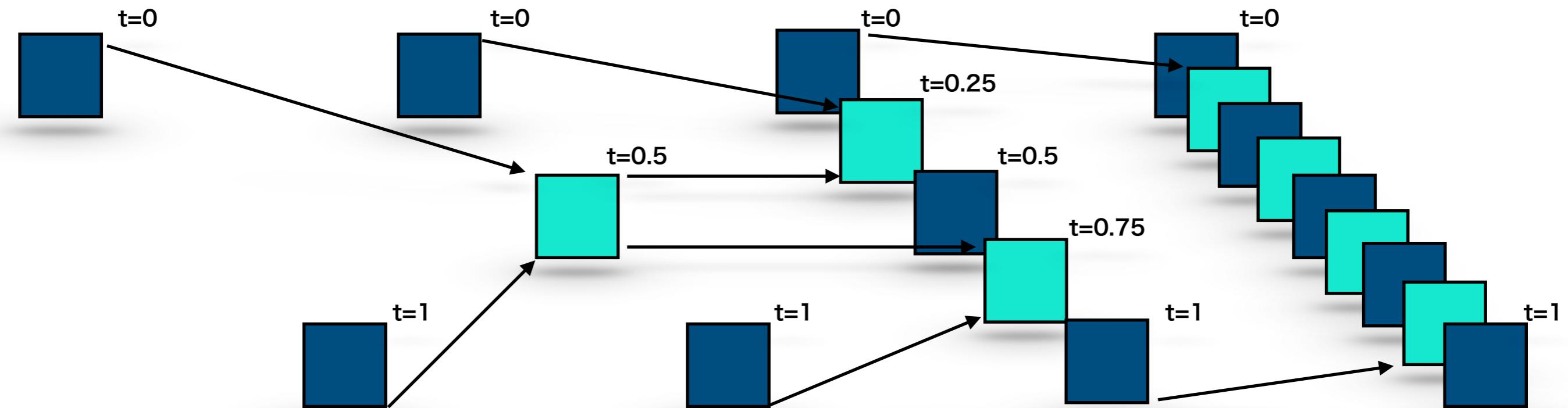
1sec

# 目次

- 1. 背景
- 2. 既存手法
- 3. 提案手法
- 4. 評価
- 5. まとめ

# 既存手法

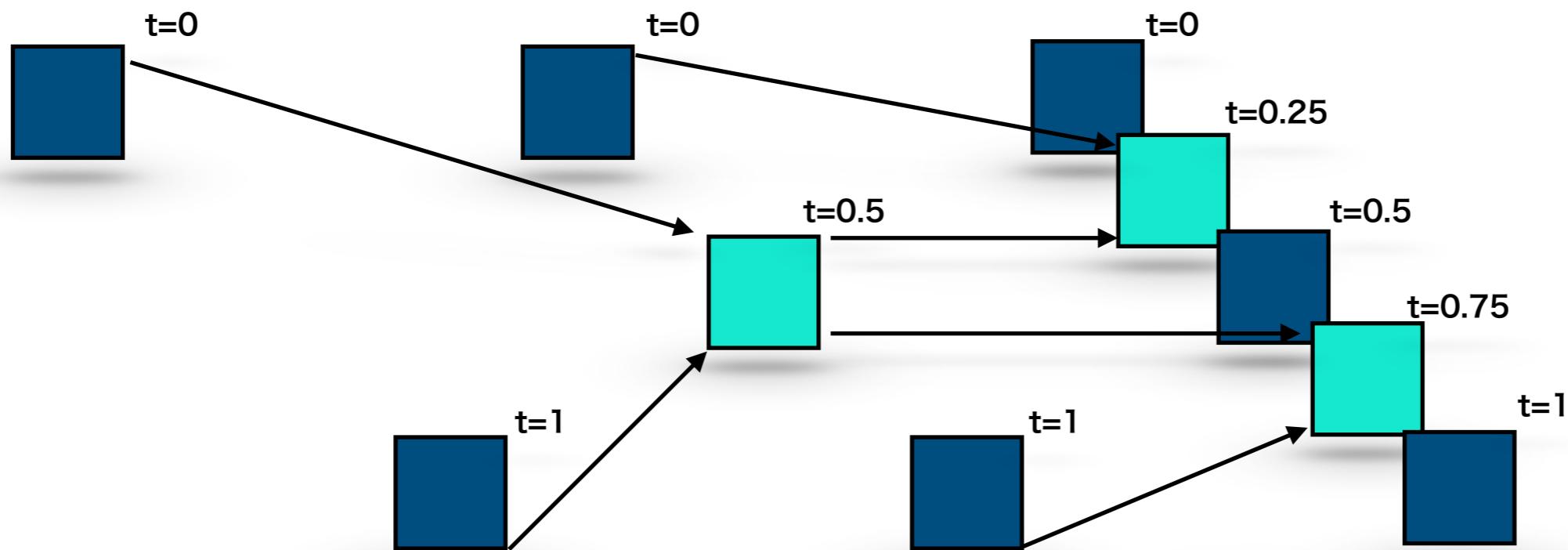
- G. Long et al, “Learning image matching by simply watching video”, In ECCV, (2016).
- D. Mahajan et al., “Moving gradients: a path-based method for plausible image interpolation”, ACM TOG, (2009).
- etc



# 既存手法の問題点

## 問題点

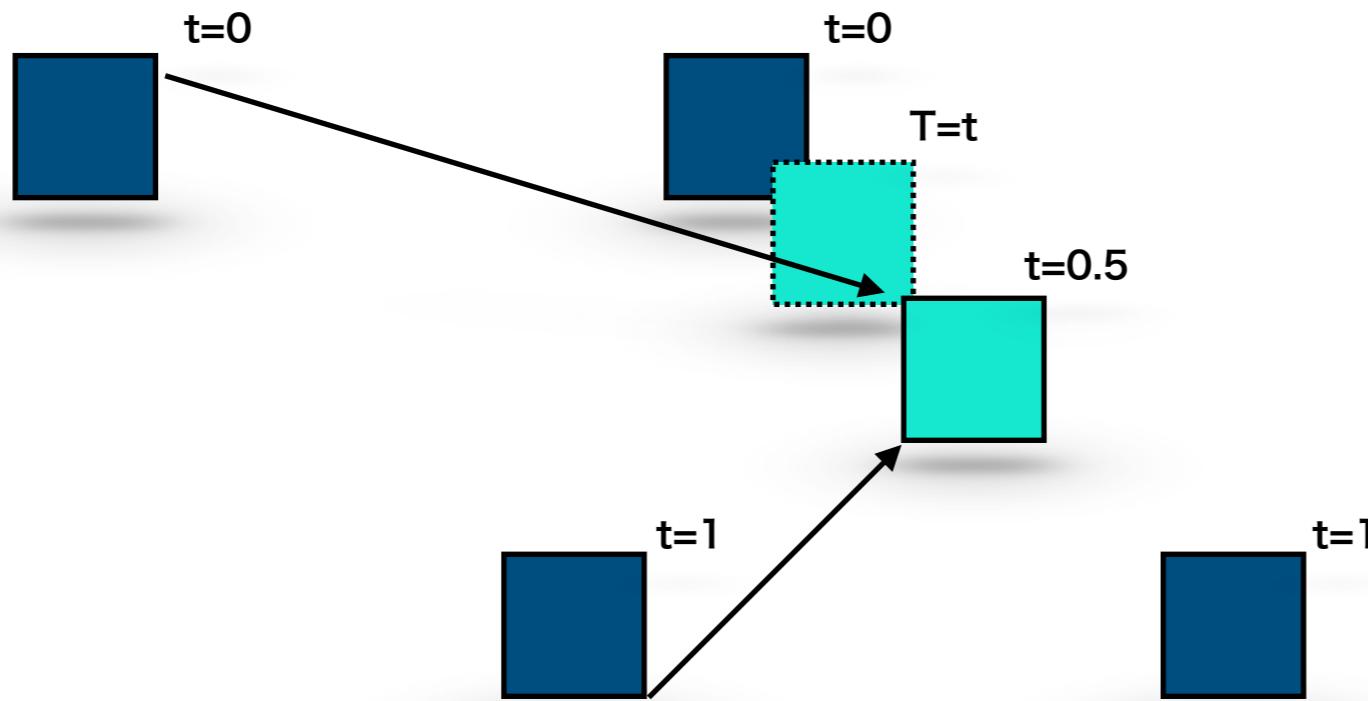
- 1.補間の過程で計算時間がかかる
- 2.補間できる枚数に制限がある
- 3.Occlusionに不適



# 目次

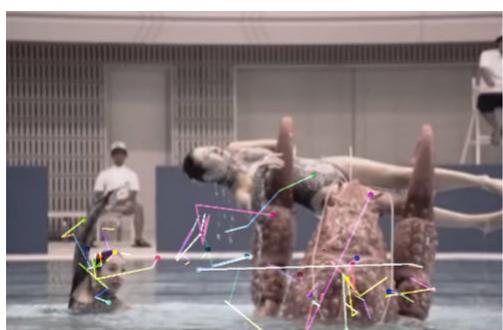
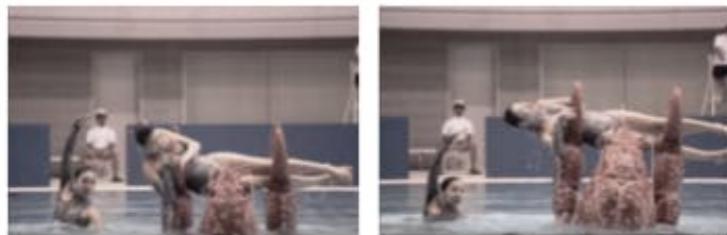
- 1. 背景
- 2. 既存手法
- 3. 提案手法
- 4. 評価
- 5. まとめ

# 提案手法



1. 任意の時間のフレーム補間を行うことで必要な枚数のフレーム補間  
→ “Optical flow” の導入  
→ FlowNet2：既存手法
2. “Visibility map” を追加して Occlusion にも適応

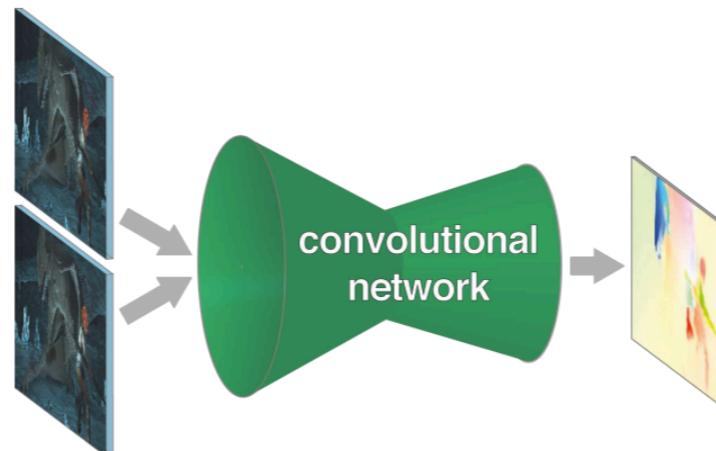
# FlowNet2



**Optical flow:**

二つの画像間での各点が  
どのように動いたかベクトルで表現

FlowNet2 で提案されてる CNN を使用

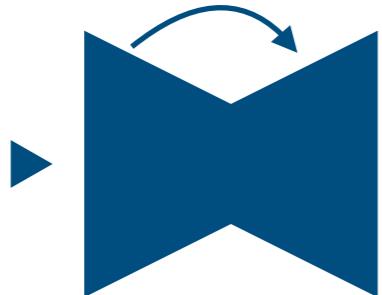


E. Ilg, et al., "Flownet 2.0: Evolution of optical flow estimation with deep networks", CVPR(2017)

# Entire Network

## Flow computation CNN

入力画像



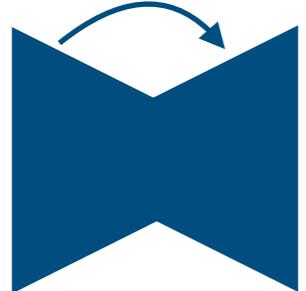
Optical  
flow

Prediction

**FlowNet2**

## Flow computation CNN

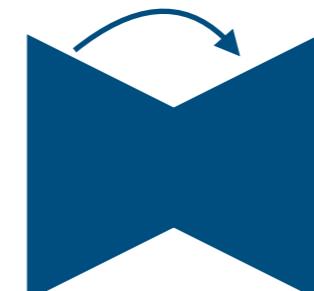
入力画像



Optical flow

## Flow interpolation CNN

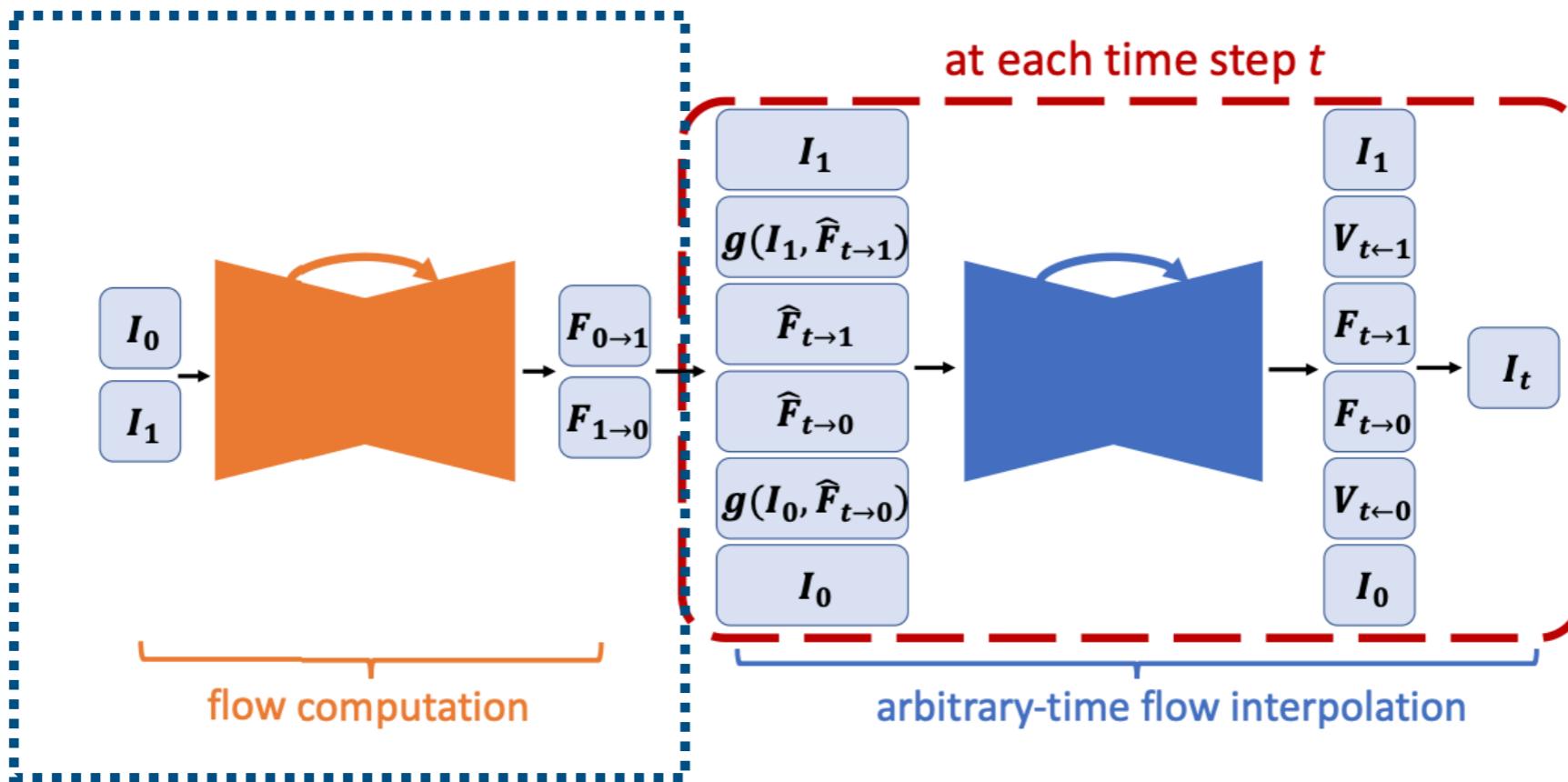
- Arbitrary Optical flow
- Bilinear interpolation



Visibility  
Map

Prediction

# Flow computation CNN

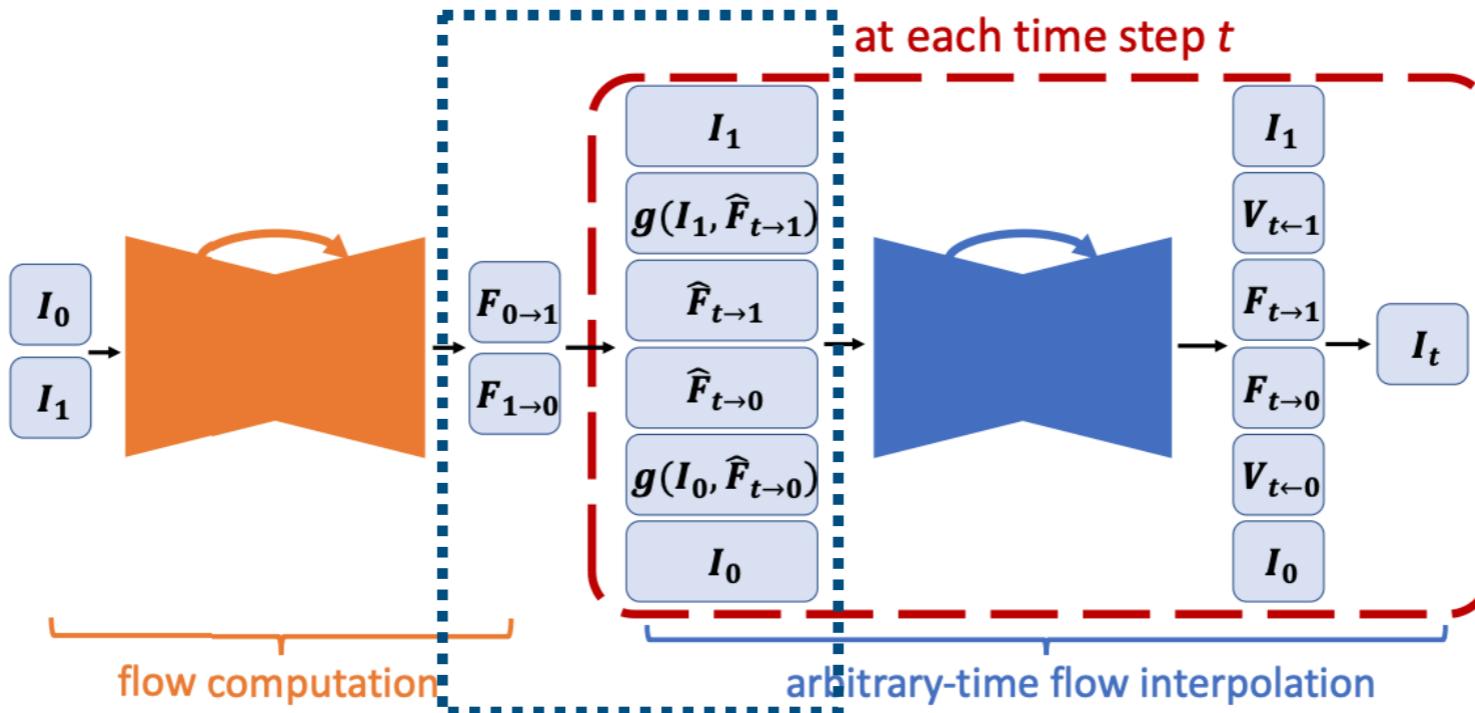


Flow computation CNN

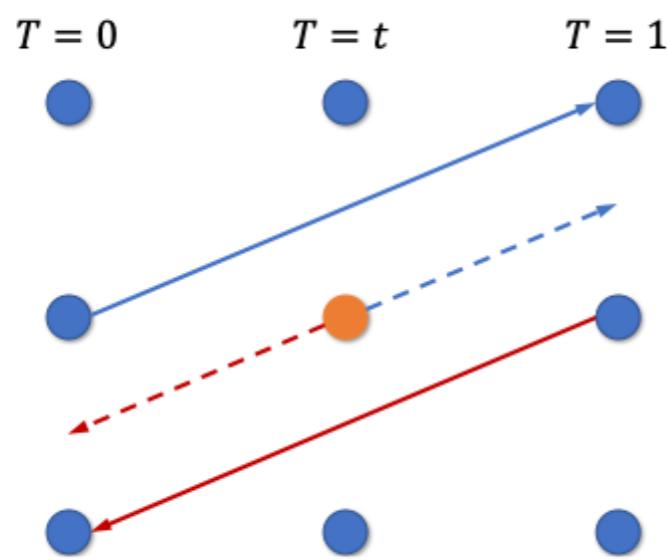
二つの入力画像,  $I_0, I_1$  (RGB)

CNNを用いて Optical flow  $F_{0 \rightarrow 1}, F_{1 \rightarrow 0}$ を推定

# Arbitrary-time Optical Flow



## Optical flow



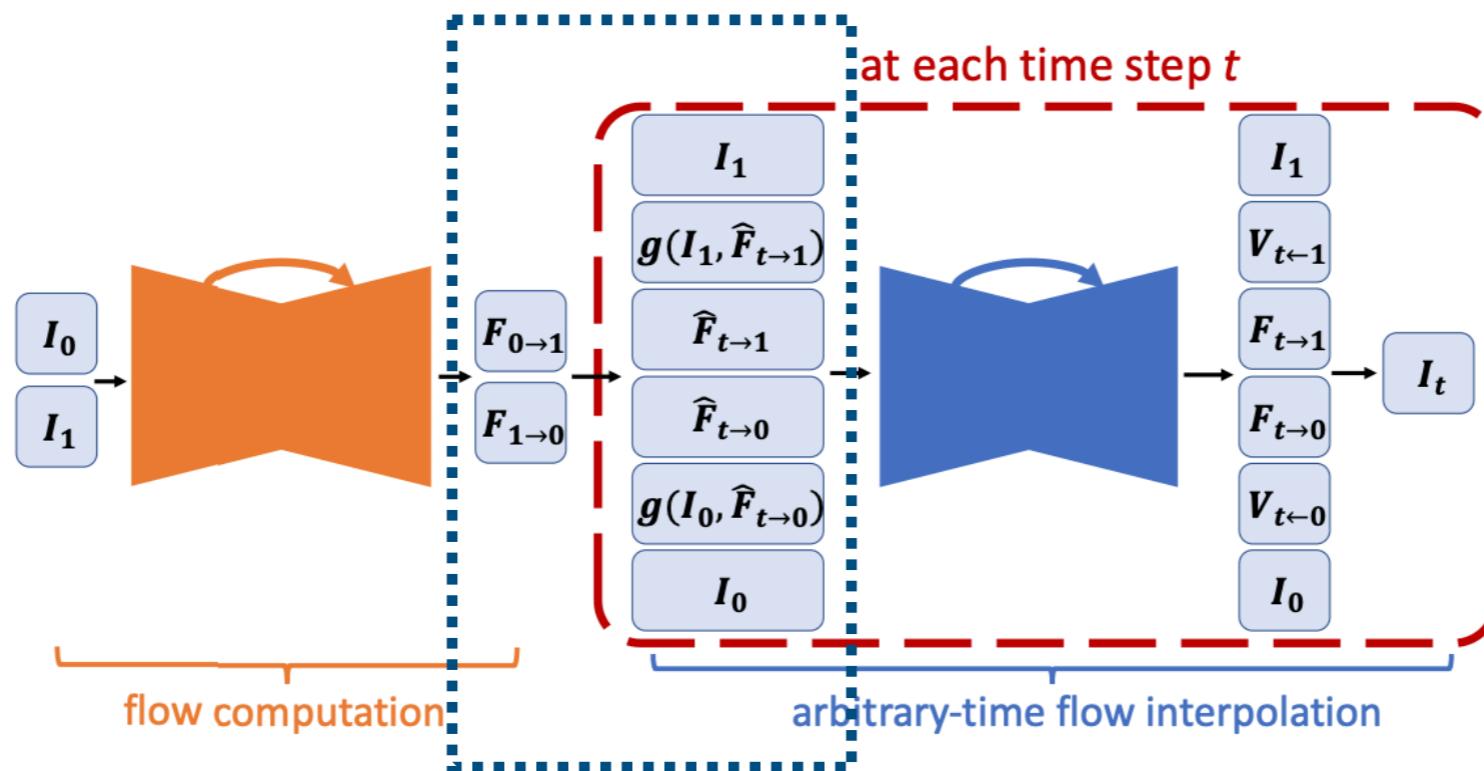
実線： $F_{0 \rightarrow 1}, F_{1 \rightarrow 0}$

入力画像間のOptical flow

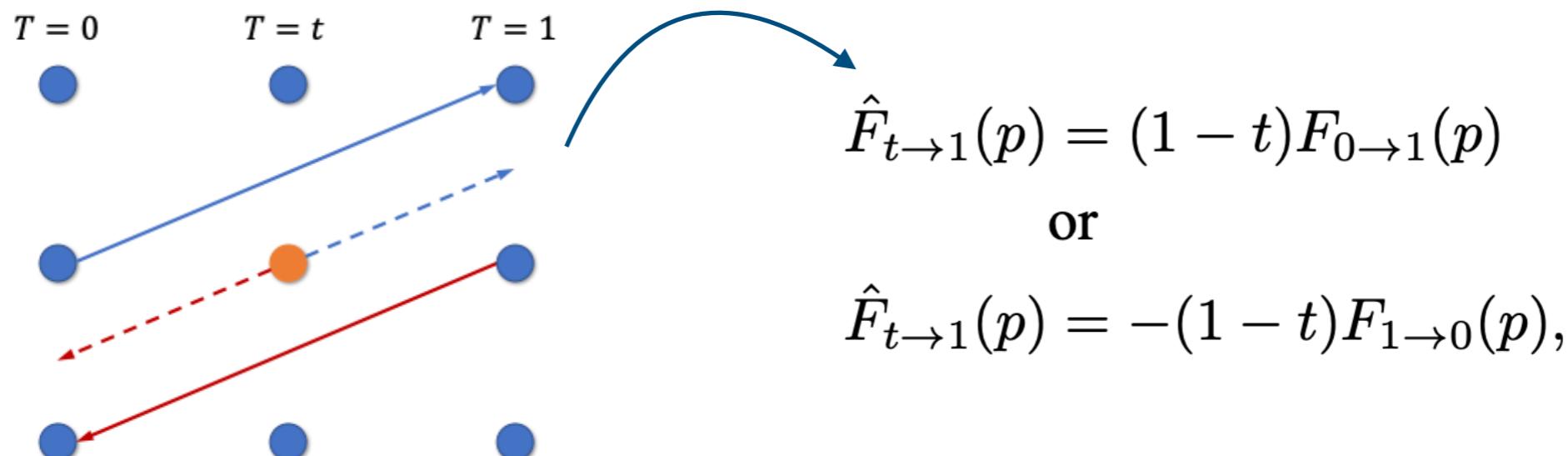
破線： $F_{t \rightarrow 1}, F_{t \rightarrow 0}$

入力画像と補間フレーム間の  
Optical flow

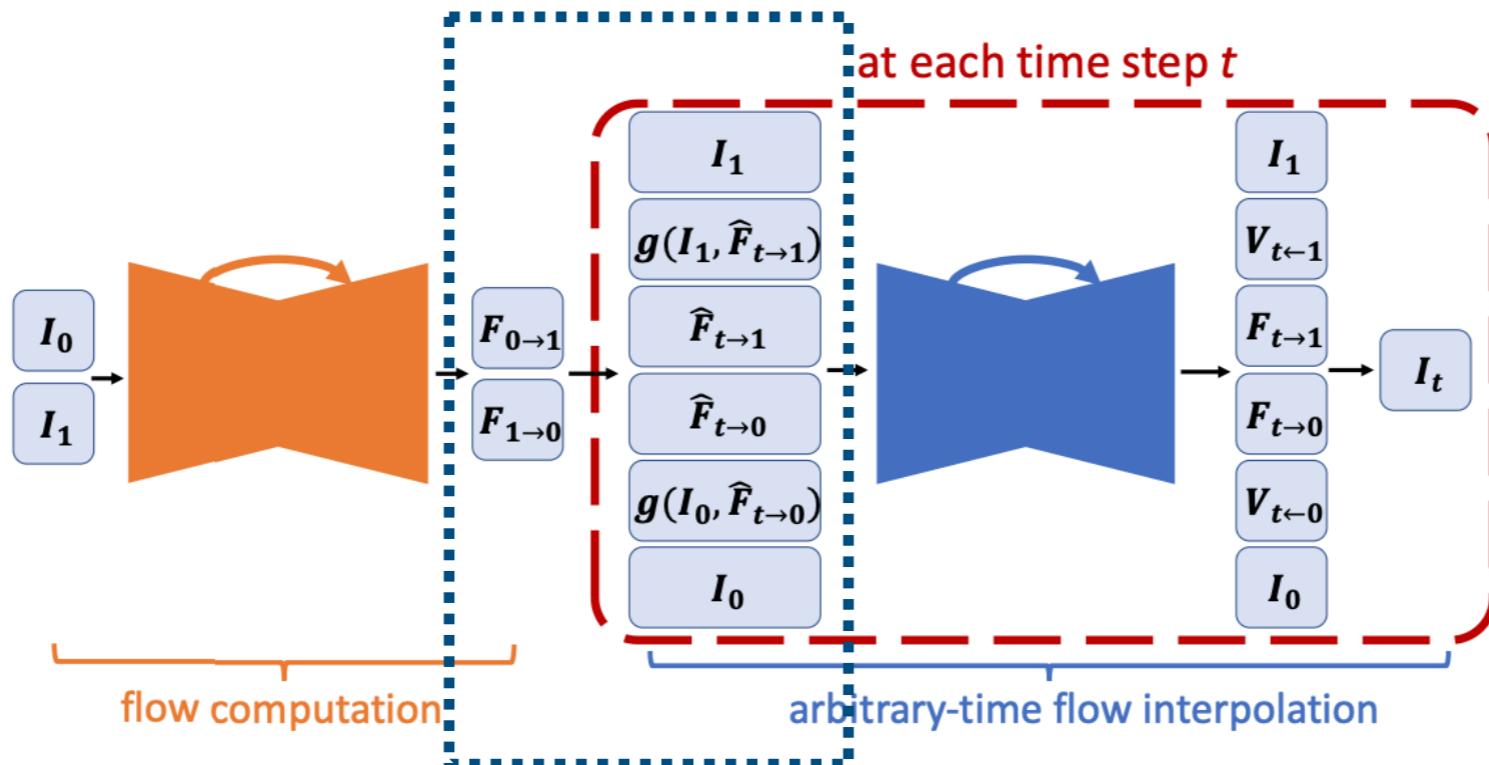
# Arbitrary-time Optical Flow 2



入力画像から補間フレームまでのOptical flow



# Bilinear Interpolation



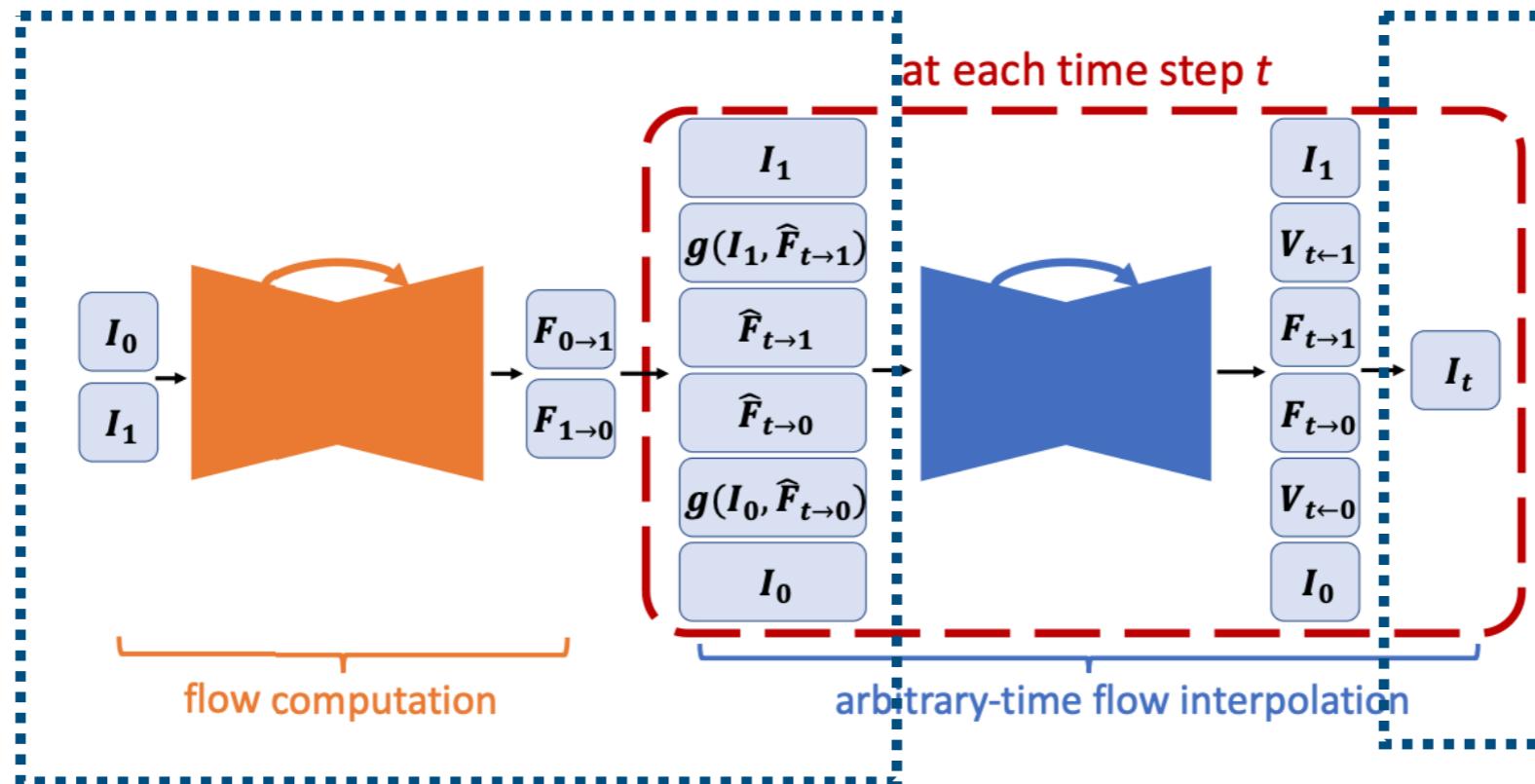
Bilinear interpolation :  $g(I, F)$

求めたい点の周辺4点から画素値を計算する関数

$g(I_0, F_{t \rightarrow 0})$  : 入力画像  $I_0$  と optical flow  $F_{t \rightarrow 0}$  により推定されたフレーム

$g(I_1, F_{t \rightarrow 1})$  : 入力画像  $I_1$  と optical flow  $F_{t \rightarrow 1}$  により推定されたフレーム

# Flow computation CNN による推定

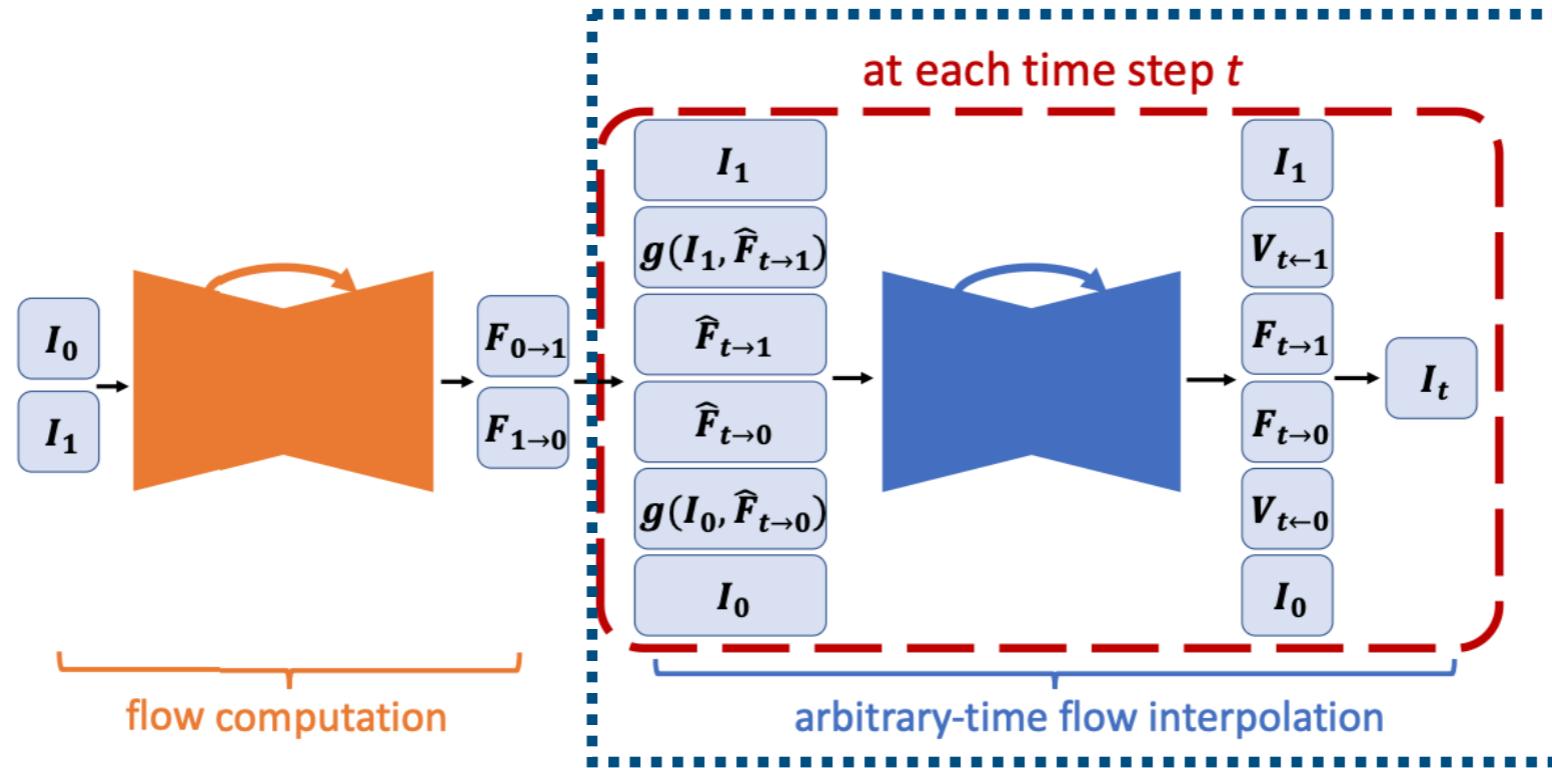


Flow computation CNN

$$\hat{I}_t = \alpha_0 \odot g(I_0, F_{t \rightarrow 0}) + (1 - \alpha_0) \odot g(I_1, F_{t \rightarrow 1}),$$

$\alpha_0$  は  $g(I_0, F_{t \rightarrow 0})$ ,  $g(I_1, F_{t \rightarrow 1})$  の依存度を表すパラメーター

# Visibility map



Flow interpolation CNN

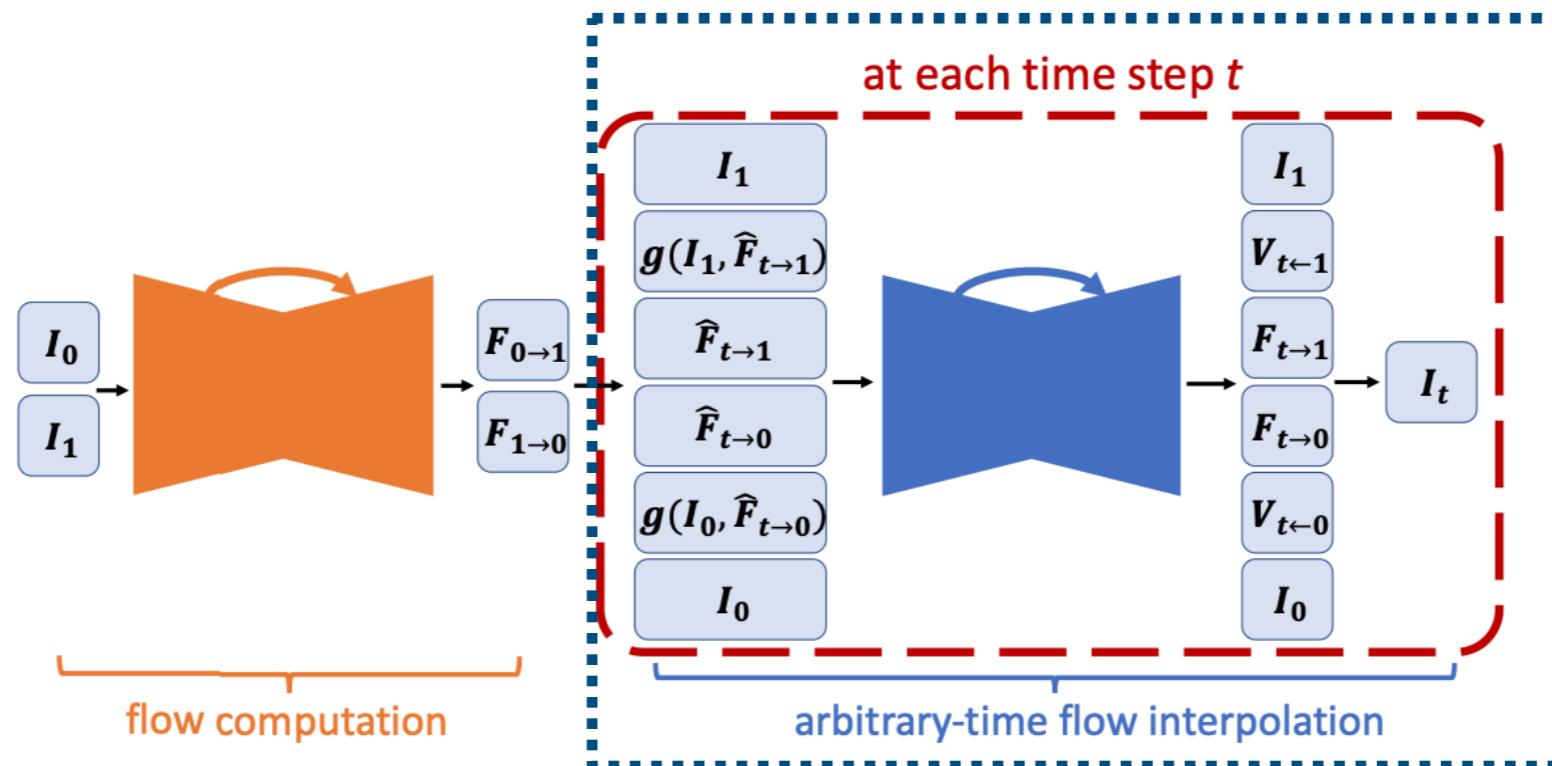
$\alpha_0$  だけではOcclusionに対応できない…

Visibility map :  $V_{t \rightarrow 0}$ ,  $V_{t \rightarrow 1}$

0~1の値で物体がどのくらい隠れているかを表す

$V_{t \leftarrow 0} = 1 - V_{t \leftarrow 1}$  の条件のもと CNNで推定

# Visibility map の導入による推定



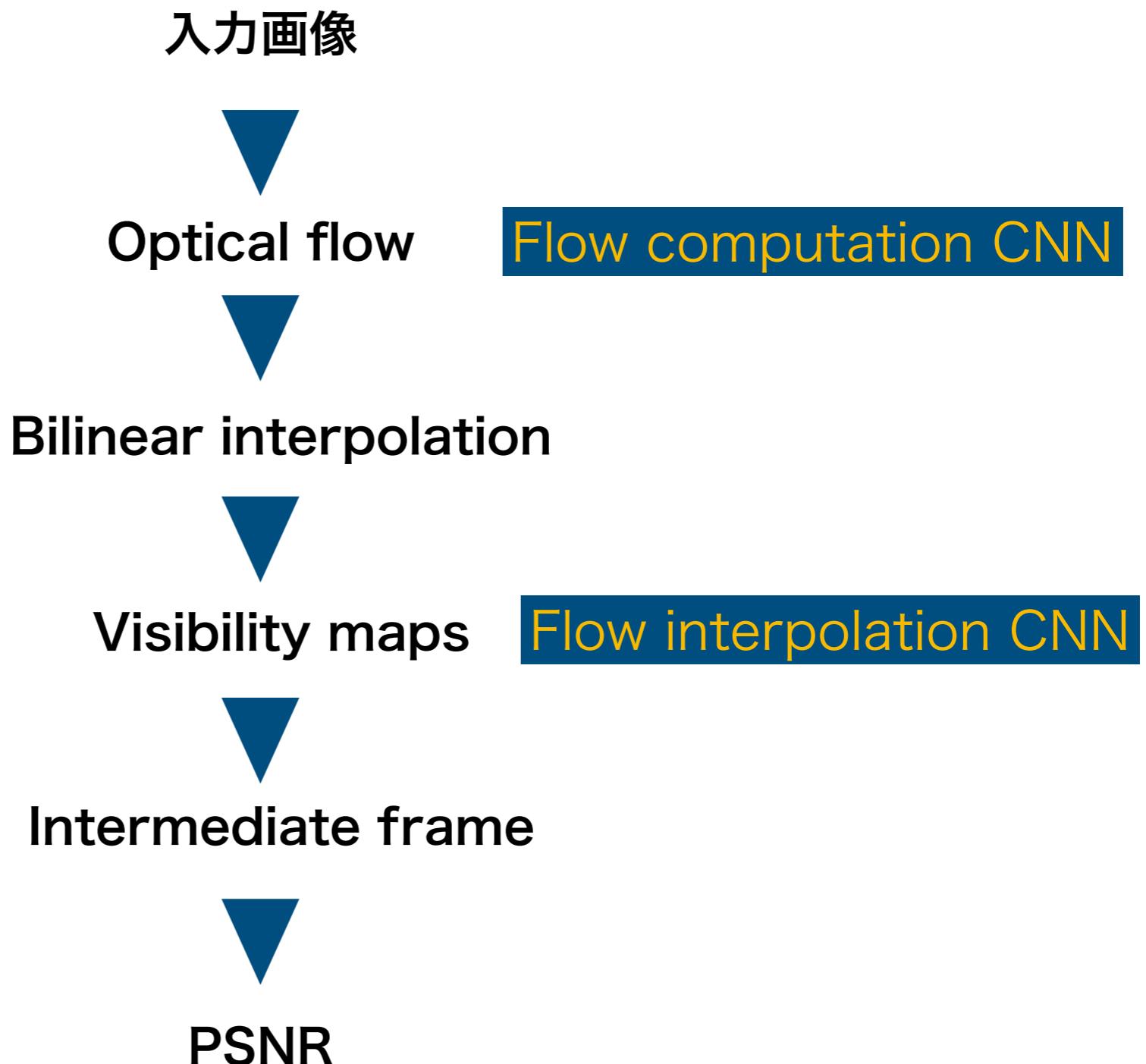
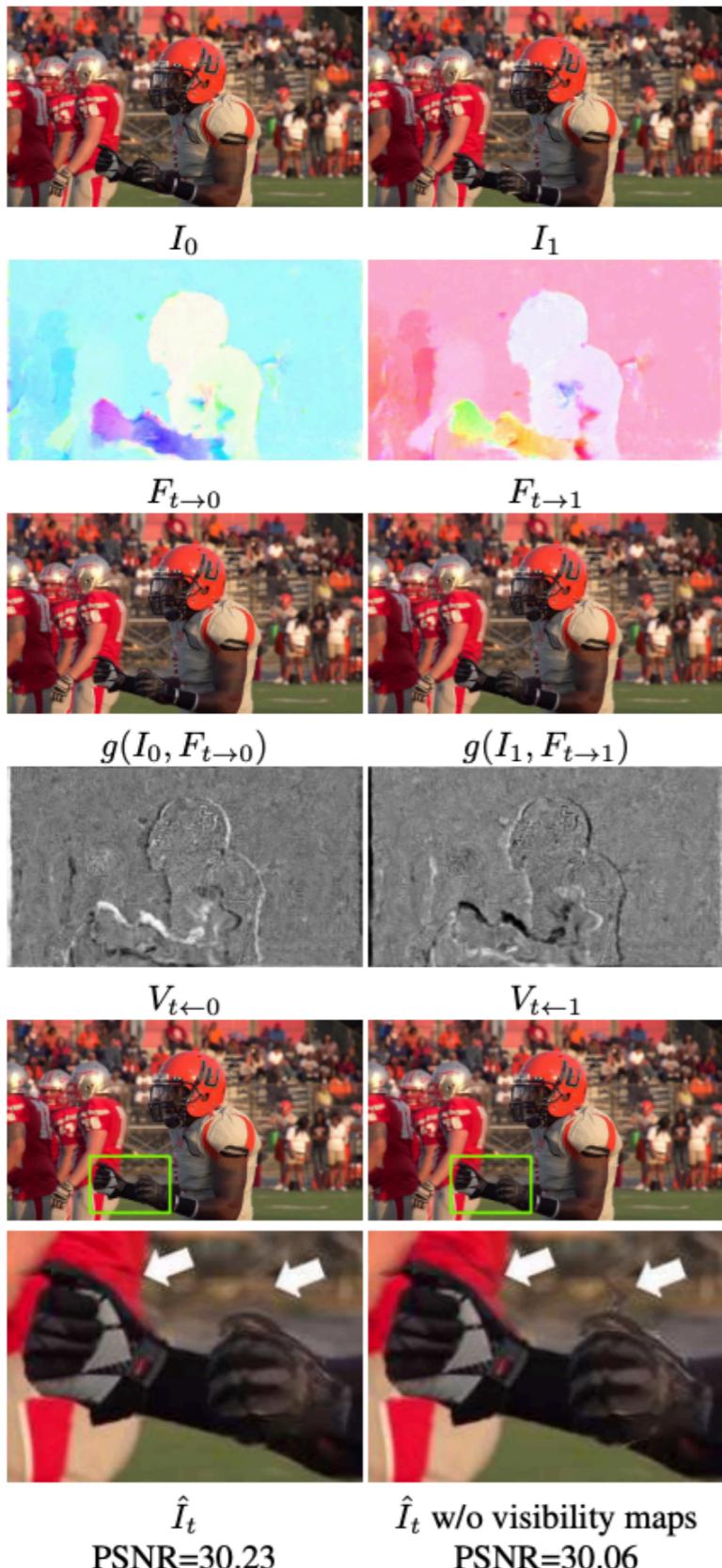
## Flow interpolation CNN

$$\alpha_0 = \frac{(1-t)}{Z} \mathbf{V}_{t \leftarrow 0} \quad 1 - \alpha_0 = \frac{t}{Z} \mathbf{V}_{t \leftarrow 1}$$

→ 
$$\hat{I}_t = \frac{1}{Z} \odot ((1-t)V_{t \leftarrow 0} \odot g(I_0, F_{t \rightarrow 0}) + tV_{t \leftarrow 1} \odot g(I_1, F_{t \rightarrow 1})),$$

where  $Z = (1-t)V_{t \leftarrow 0} + tV_{t \leftarrow 1}$  is a normalization factor.

# Samples of predicted visibility maps



# Training

$$l = \lambda_r l_r + \lambda_p l_p + \lambda_w l_w + \lambda_s l_s \quad : \text{損失関数}$$

Reconstruction loss     $l_r$  : 補間されたフレーム(画素)の損失

Perceptual loss     $l_p$  : VGG16を用いて補間フレーム(画素)の損失

Warping loss     $l_w$  : Bilinear interpolation による補間フレームの損失

Smoothness loss     $l_s$  : 隣り合うピクセルのoptical flow は近い値をとる

# 目次

- 1. 背景
- 2. 既存手法と問題点
- 3. 提案手法
- 4. 評価
- 5. まとめ

# 評価

Dataset:

240fps

- hand- held camera
- youtube



1,132 video clips , 376K frames



Train : 107 videos  
Test : 12 video



指標

PSNR : 信号の最大値とノイズの比

SSIM : 人間が感じる違いを指標化

IE (RMS): 誤差の二乗平均平方根

# 評価

## 1, 補間フレーム数の変化による影響

	PSNR	SSIM	IE
1 interp	30.26	0.909	8.85
3 interp	31.02	0.917	8.43
7 interp	<b>31.19</b>	<b>0.918</b>	<b>8.30</b>

## 2, 損失関数のパラメーターの必要性

	PSNR	SSIM	IE
w/o flow interpolation	30.34	0.908	8.93
w/o vis map	31.16	0.918	8.33
w/o perceptual loss	30.96	0.916	8.50
w/o warping loss	30.52	0.910	8.80
w/o smoothness loss	<b>31.19</b>	<b>0.918</b>	<b>8.26</b>
full model	<b>31.19</b>	<b>0.918</b>	8.30

**PSNR :**

信号の最大値とノイズの比

**SSIM :**

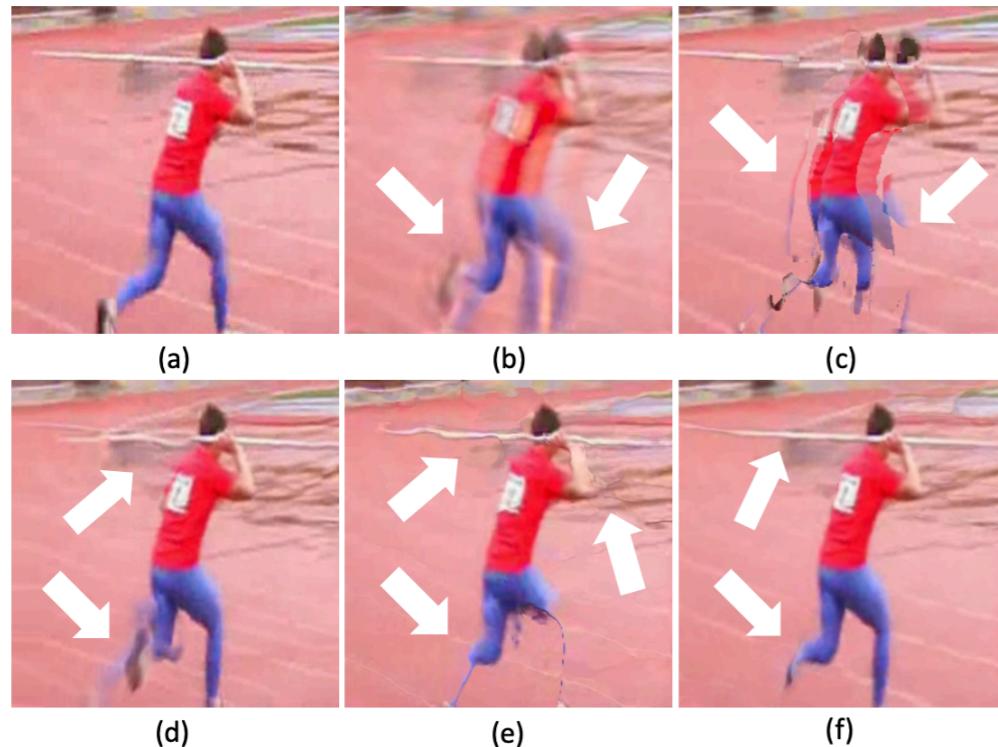
人間が感じる違いを指標化

**IE :**

RMS 誤差の二乗平均平方根

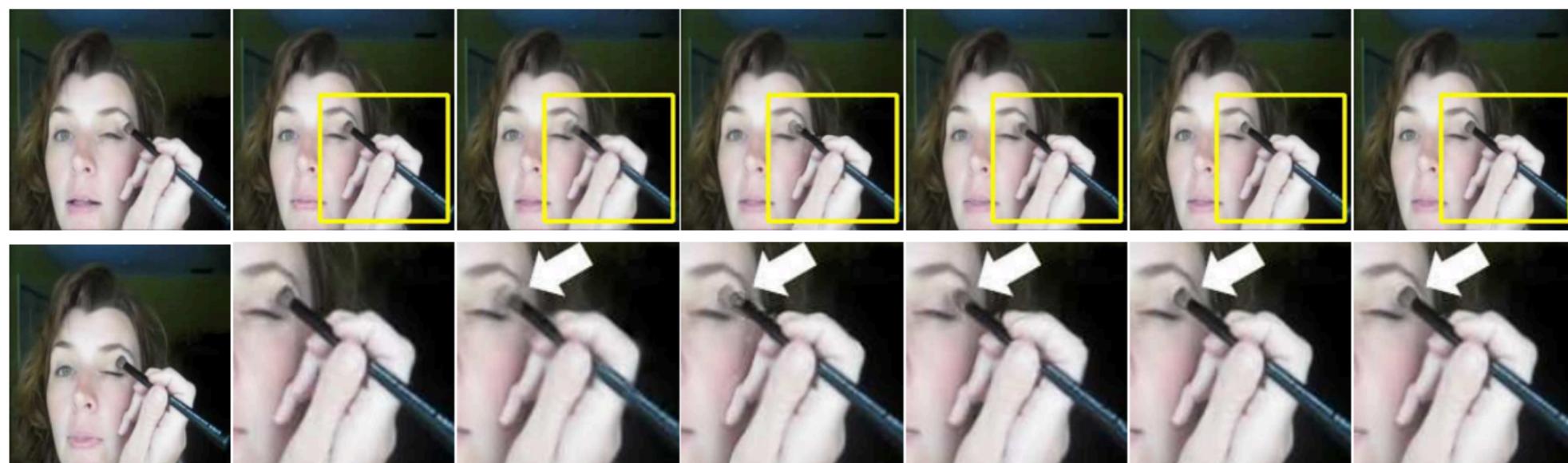
# 評価

## CNNベースの既存研究との比較



(b) Fasebased  
(c) FlowNet  
(d) SepConv  
(e) DVF  
(f) Ours

Single frame  
Interpolation



2 input images actual in-between PhaseBased [18] FlowNet2 [1, 9] DVF [15] SepConv [20] ours

Occlusion  
Reasonig

# 目次

- 1. 背景
- 2. 既存手法と問題点
- 3. 提案手法
- 4. 評価
- 5. まとめ

# まとめ

- ・二つの入力画像から必要な枚数のフレーム数を補間
- ・評価からvisibility map の必要性
- ・今回の提案手法は既存技術を上回った



# Supplementary

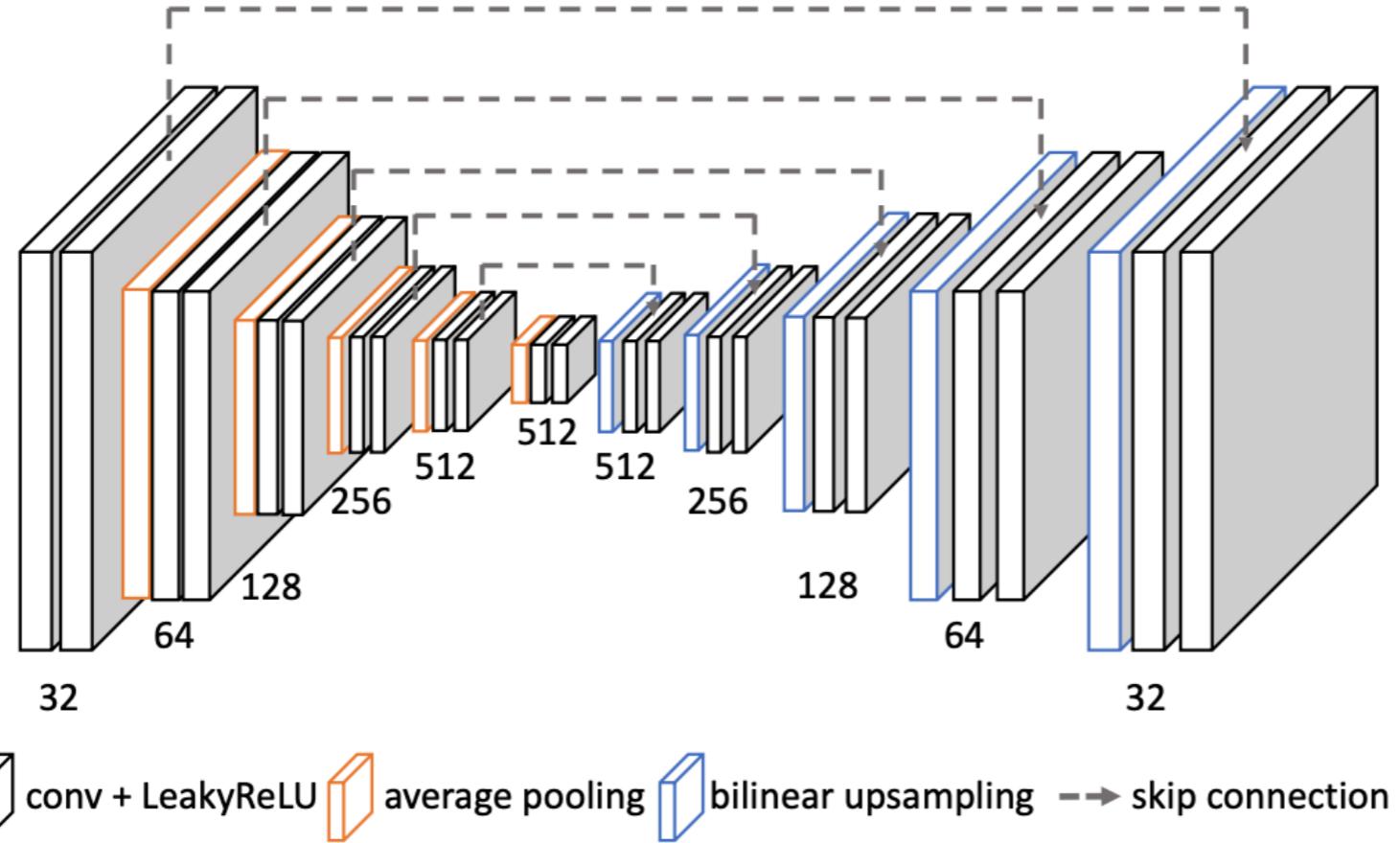
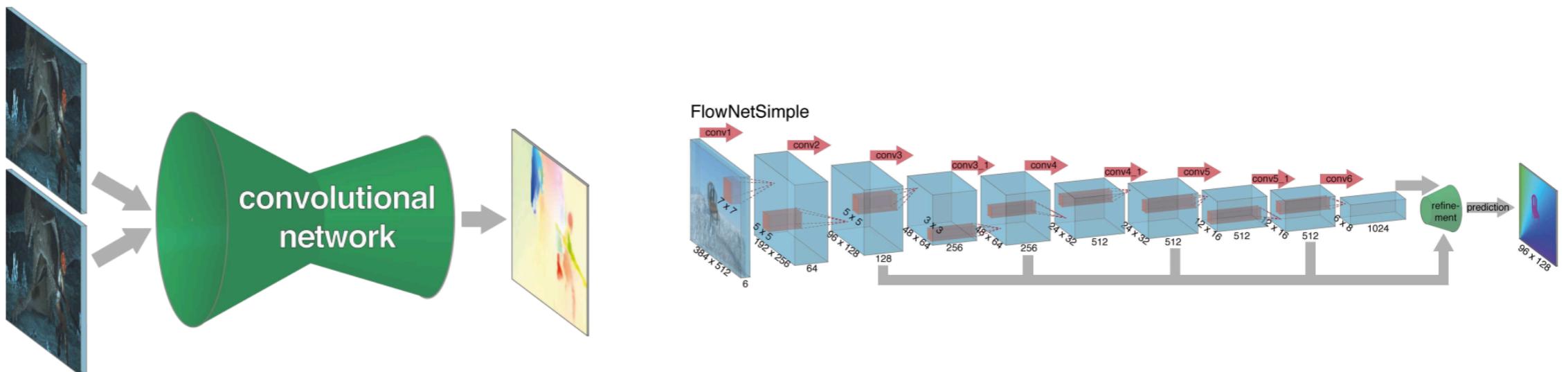
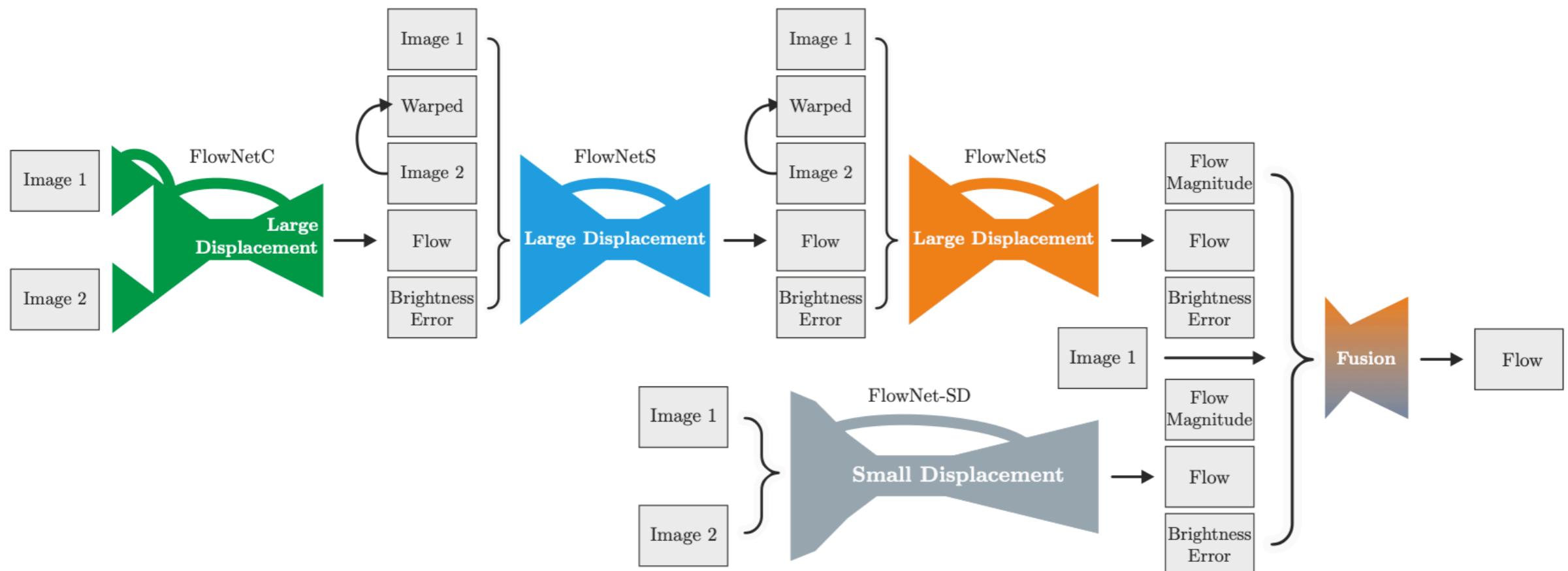


Figure 1: Illustration of the architecture of our flow computation and flow interpolation CNNs.



# Supplementary

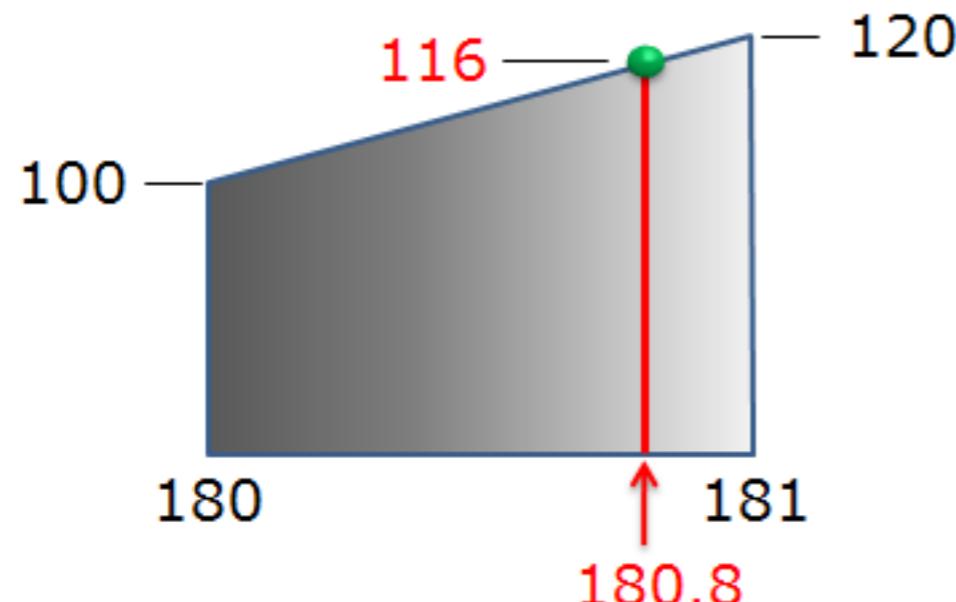


# Supplementary

バイリニア補間では求める位置(x,y)の周辺の $2 \times 2$ 画素（4画素）を使って、輝度値を直線的に補間して、輝度値を求めます。

直線的に補間する方法は、たとえば下図のようにX座標が180と181の2点間の180.8の位置の輝度値を求めるのは、

2点の位置の輝度値の比から簡単に求まります。



$$116 = 0.2 \times 100 + 0.8 \times 120$$