Chapter x
Video super resolution techniques

Variational Methods for Computer Vision
WS 16/17

Jonas Geiping
Visual Scene Analysis
Department of Computer Science
University of Siegen

Visual
Scene
Analysis

Video Super Resolution

Do you remember the image zoom from chapter 2 ?

Image zooming

Forward model = Given the high resolution version, how would you create a low resolution version?

Image zooming

Forward model = Given the high resolution version, how would you create a low resolution version?

Steps:

1. Blur the high resolution image (to avoid aliasing)
2. Average the values of the high resolution pixels to obtain the low resolutino pixel value.

Image zooming

Forward model = Given the high resolution version, how would you create a low resolution version?

Steps:

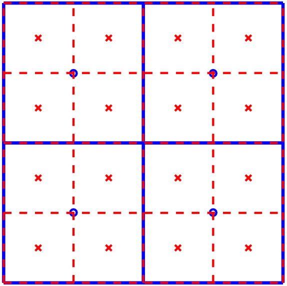
1. Blur the high resolution image (to avoid aliasing)
2. Average the values of the high resolution pixels to obtain the low resolutino pixel value.

Define $A = D\ B$ for a blur operator B and a downsampling operator D and solve

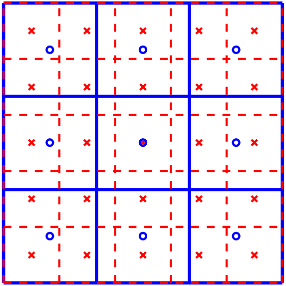
$$\frac{1}{2}\|Au - f\|^2 + \alpha R(u)$$

The downsampling operator

The downsampling procedure:



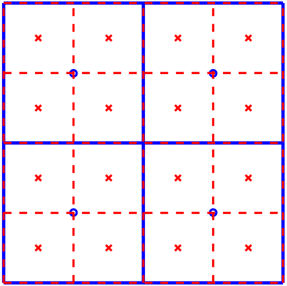
Interpolation 4×4 to 2×2

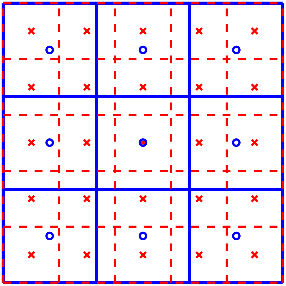


Interpolation 5×5 to 3×3

The downsampling operator

The downsampling procedure:





Interpolation 4×4 to 2×2 Interpolation 5×5 to 3×3

Possible approach to generate a forward model: The values at the blue (low resolution) pixels originate from the red (high resolution) pixels via a bilinear interpolation.

Problem: Information that is lost by the downsampling can never be recovered.

Problem: Information that is lost by the downsampling can never be recovered. We can only connect the existing information according to the chosen regularizer:
For example:

Problem: Information that is lost by the downsampling can never be recovered. We can only connect the existing information according to the chosen regularizer:
For example:

$$\frac{1}{2}\|Au - f\|^2 + \alpha||\nabla u||_2^2$$

Problem: Information that is lost by the downsampling can never be recovered. We can only connect the existing information according to the chosen regularizer:
For example:

$$\frac{1}{2}\|Au - f\|^2 + \alpha||\nabla u||_2^2$$

(smoothly)

Problem: Information that is lost by the downsampling can never be recovered. We can only connect the existing information according to the chosen regularizer:
For example:

$$\frac{1}{2}\|Au - f\|^2 + \alpha||\nabla u||_2^2$$

(smoothly)

$$\frac{1}{2}\|Au - f\|^2 + \alpha||\nabla u||_1$$

Problem: Information that is lost by the downsampling can never be recovered. We can only connect the existing information according to the chosen regularizer:
For example:

$$\frac{1}{2}\|Au - f\|^2 + \alpha||\nabla u||_2^2$$

(smoothly)

$$\frac{1}{2}\|Au - f\|^2 + \alpha||\nabla u||_1$$

(piecewise constant)

However what if we have a video in a low resolution ?

Low Resolution Frames

Low Resolution Frames

Low Resolution Frames

Low Resolution Frames

Low Resolution Frames

Video Super Resolution

- We still do not know the details in every single image, but we now have several shots of the same object.

[1]This is also why MPEG compression schemes work so well, they also rely on motion estimators.

Video Super Resolution

- We still do not know the details in every single image, but we now have several shots of the same object.
- Remember that videos are often 24fps or more, so there is a lot of repetition [1]

[1] This is also why MPEG compression schemes work so well, they also rely on motion estimators.

Video Super Resolution

- We still do not know the details in every single image, but we now have several shots of the same object.
- Remember that videos are often 24fps or more, so there is a lot of repetition [1]

[1] This is also why MPEG compression schemes work so well, they also rely on motion estimators.

Video Super Resolution

- We still do not know the details in every single image, but we now have several shots of the same object.
- Remember that videos are often 24fps or more, so there is a lot of repetition [1]

In which situations do we gain information ?

[1] This is also why MPEG compression schemes work so well, they also rely on motion estimators.

Video Super Resolution

- We still do not know the details in every single image, but we now have several shots of the same object.
- Remember that videos are often 24fps or more, so there is a lot of repetition [1]

In which situations do we gain information ?

- If an image object is not moving at all, then there is no new information

[1] This is also why MPEG compression schemes work so well, they also rely on motion estimators.

Video Super Resolution

- We still do not know the details in every single image, but we now have several shots of the same object.
- Remember that videos are often 24fps or more, so there is a lot of repetition [1]

In which situations do we gain information ?

- If an image object is not moving at all, then there is no new information
- If an object is moving on (or close to) the low-resolution grid, there is no new information.

[1] This is also why MPEG compression schemes work so well, they also rely on motion estimators.

Video Super Resolution

- We still do not know the details in every single image, but we now have several shots of the same object.
- Remember that videos are often 24fps or more, so there is a lot of repetition [1]

In which situations do we gain information ?

- If an image object is not moving at all, then there is no new information
- If an object is moving on (or close to) the low-resolution grid, there is no new information.
- $\Rightarrow$ Only moving objects have new information

[1] This is also why MPEG compression schemes work so well, they also rely on motion estimators.

Motion Estimation

However, the new information we need is locations at different space/time coordinates in the video, which we do not know in general!

Motion Estimation

However, the new information we need is locations at different space/time coordinates in the video, which we do not know in general!

We need to estimate the motion between video frames, to 'register' common objects.

Motion Estimation

However, the new information we need is locations at different space/time coordinates in the video, which we do not know in general!

We need to estimate the motion between video frames, to 'register' common objects.

We need to do this at precisely as possible, at minimum with a higher precision than the low-resolution grid.

Example Motion Estimation: Optical flow

Similar to stereo vision, we try to find the vector field $v(x)$ (x is know 2-dimensional !) that maps the intensities of two images f_1 and f_2:

$$f_1(x + v(x)) \approx f_2(x) \tag{1}$$

Michael will go into further details next week.

Example Motion Estimation: Optical flow

Similar to stereo vision, we try to find the vector field $v(x)$ (x is know 2-dimensional !) that maps the intensities of two images f_1 and f_2:

$$f_1(x + v(x)) \approx f_2(x) \tag{1}$$

Michael will go into further details next week.

Note however that moving an image by a *known* vector field v is a linear operation (as we are 'just' moving and interpolating pixels), so we can write

$$f_1(x + v(x)) \approx (Wf_1)(x). \tag{2}$$

We call this matrix the 'warp operator' to the motion 'v'. It is inexact in finite dimensions as it interpolates unknown positions.

Example Motion Estimation: Optical flow

Example Motion Estimation: Optical flow

Example Motion Estimation: Optical flow

Multiframe Motion Coupling

Previously we talked about ”information” in very general terms. But how do we define a variational model for our video super resolution ?

Variational model

Extending our previous zooming to n frames we can write

$$\sum_{i=1}^{n} ||DBu_i - f_i||_1 + \alpha \sum_{i=1}^{n} ||\nabla u_i||_1 \qquad (3)$$

Variational model

Extending our previous zooming to n frames we can write

$$\sum_{i=1}^{n} ||DBu_i - f_i||_1 + \alpha \sum_{i=1}^{n} ||\nabla u_i||_1 \tag{3}$$

I replaced the previous L2-norm with an L1 norm, as this is experimentally more robust against outliers.
Otherwise there is nothing new here.

Variational model

Extending our previous zooming to n frames we can write

$$\sum_{i=1}^{n} ||DBu_i - f_i||_1 + \alpha \sum_{i=1}^{n} ||\nabla u_i||_1 \tag{3}$$

I replaced the previous L2-norm with an L1 norm, as this is experimentally more robust against outliers.
Otherwise there is nothing new here.

$\Rightarrow$ Where do we place our warp operator ?

Single-frame coupling

A classical approach is to add the additional images as additional data terms:
In addition to the old data relation

$$DBu_i = f_i$$

Single-frame coupling

A classical approach is to add the additional images as additional data terms:
In addition to the old data relation

$$DBu_i = f_i$$

we also demand

$$DBW_{ij}u_i = f_i \qquad \forall j \in [1, \dots, n]$$

where W_{ij} is the warp operator that 'moves' the i-th frame to the j-th frame.

Single-frame coupling

A classical approach is to add the additional images as additional data terms:
In addition to the old data relation

$$DBu_i = f_i$$

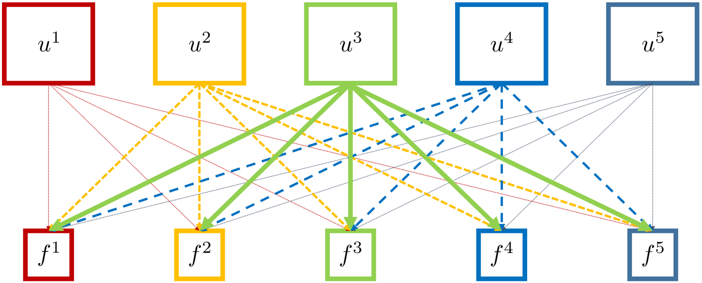
we also demand

$$DBW_{ij}u_i = f_i \qquad \forall j \in [1, \dots, n]$$

where W_{ij} is the warp operator that 'moves' the i-th frame to the j-th frame.
In total we get:

$$\sum_{i=1}^{n} \sum_{j \neq i}^{n} ||DBW_{ij}u_i - f_i||_1 + \alpha \sum_{i=1}^{n} ||\nabla u_i||_1$$

Single-frame coupling



Single-frame coupling

This approach has several issues:

Single-frame coupling

This approach has several issues:

- The new high-resolution video $u_1, \ldots, u_n$ is not coupled directly. Differences in motion error between frames can lead to jittering and other inconsistencies in the video.

Single-frame coupling

This approach has several issues:

- The new high-resolution video $u_1, \ldots, u_n$ is not coupled directly. Differences in motion error between frames can lead to jittering and other inconsistencies in the video.
- To couple all n frames in a video we need to compute $n(n-1)$ flow computations.

Single-frame coupling

This approach has several issues:

- The new high-resolution video $u_1, \ldots, u_n$ is not coupled directly. Differences in motion error between frames can lead to jittering and other inconsistencies in the video.
- To couple all n frames in a video we need to compute $n(n-1)$ flow computations.
- The motion between frames that are several seconds apart is often much greater than between subsequent frames (which are only $\frac{1}{24} = 0.04$ seconds apart) and as such much harder to estimate.

Multi-frame motion coupling

Idea: Instead of demanding from our minimizer that

$$DBW_{ij}u_i = f_i \qquad \forall j \in [1, \ldots, n]$$

we demand

$$W_{i,i+1}u_i = u_{i+1} \qquad \forall i \in [1, \ldots, n]$$

Multi-frame motion coupling

Idea: Instead of demanding from our minimizer that

$$DBW_{ij}u_i = f_i \qquad \forall j \in [1, \ldots, n]$$

we demand

$$W_{i,i+1}u_i = u_{i+1} \qquad \forall i \in [1, \ldots, n]$$

$\Rightarrow$ Every frame in the output video is now coupled to the next frame. The connection to all other frames is implicit.

Multi-frame motion coupling

Idea: Instead of demanding from our minimizer that

$$DBW_{ij}u_i = f_i \qquad \forall j \in [1, \dots, n]$$

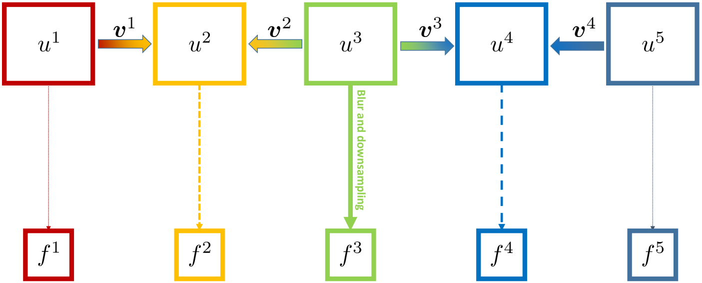
we demand

$$W_{i,i+1}u_i = u_{i+1} \qquad \forall i \in [1, \dots, n]$$

$\Rightarrow$ Every frame in the output video is now coupled to the next frame. The connection to all other frames is implicit.

$$\sum_{i=1}^{n} ||DBu_i - f_i||_1 + \alpha_1 \sum_{i=1}^{n-1} ||W_{i,i+1}u_i - u_{i+1}||_1 + \alpha_2 \sum_{i=1}^{n} ||\nabla u_i||_1$$

Multi-frame motion coupling



Multi-frame motion coupling

Advantages of this approach:

Multi-frame motion coupling

Advantages of this approach:

- The new high-resolution video $u_1, \dots, u_n$ is coupled directly. Differences in motion error between frames are minimized.

Multi-frame motion coupling

Advantages of this approach:

- The new high-resolution video $u_1, \ldots, u_n$ is coupled directly. Differences in motion error between frames are minimized.
- To couple all n frames in a video we need to compute $(n-1)$ flow computations.

Multi-frame motion coupling

Advantages of this approach:

- The new high-resolution video $u_1, \ldots, u_n$ is coupled directly. Differences in motion error between frames are minimized.
- To couple all n frames in a video we need to compute $(n-1)$ flow computations.
- The motion between subsequent frames (which are only $\frac{1}{24} = 0.04$ seconds apart) is easy to estimate.

Parameter choice with imprecise optical flow

A general problem remains:
How do we choose α_1 and α_2 ?

Parameter choice with imprecise optical flow

A general problem remains:
How do we choose α_1 and α_2 ?

Considerations:

- Increasing α_1 increases the reliance on the warp operator

Parameter choice with imprecise optical flow

A general problem remains:
How do we choose α_1 and α_2 ?

Considerations:

- Increasing α_1 increases the reliance on the warp operator
  - More details if warp is precise

Parameter choice with imprecise optical flow

A general problem remains:
How do we choose α_1 and α_2 ?

Considerations:

- Increasing α_1 increases the reliance on the warp operator
  - More details if warp is precise
  - Spatial errors if warp is imprecise

Parameter choice with imprecise optical flow

A general problem remains:
How do we choose α_1 and α_2 ?

Considerations:

- Increasing α_1 increases the reliance on the warp operator
  - More details if warp is precise
  - Spatial errors if warp is imprecise
- Increasing α_2 increases spatial coherence at the cost of detail levels.

Parameter choice with imprecise optical flow

A general problem remains:
How do we choose α_1 and α_2 ?

Considerations:

- Increasing α_1 increases the reliance on the warp operator
  - More details if warp is precise
  - Spatial errors if warp is imprecise
- Increasing α_2 increases spatial coherence at the cost of detail levels.
- Increasing both α_1 and α_2 decreases the coherence to the actual data.

Infimal Convolution regularization

A way to improve the parameter choice is to switch from additive regularizers to infimal convolution.
Remember chapter 2 ?

Infimal Convolution regularization

A way to improve the parameter choice is to switch from additive regularizers to infimal convolution.
Remember chapter 2 ?

$$(R_1 \Box R_2)(u) = \min_w R_1(u - w) + R_2(w)$$

Infimal Convolution regularization

A way to improve the parameter choice is to switch from additive regularizers to infimal convolution.
Remember chapter 2 ?

$$(R_1 \Box R_2)(u) = \min_w R_1(u - w) + R_2(w)$$

Let us define

$$R_{temp}(u) = \sum_{i=1}^{n-1} ||W_{i,i+1} u_i - u_{i+1}||_1$$

and

$$R_{spat}(u) = \sum_{i=1}^{n} ||\nabla u_i||_1$$

Infimal Convolution regularization

Now we have a new regularizer

$$R(u) = \min_{w} R_{temp}(u - w) + R_{spat}(w)$$

Infimal Convolution regularization

Now we have a new regularizer

$$R(u) = \min_{w} R_{temp}(u - w) + R_{spat}(w)$$

Interpretation:

- If the warp is imprecise (and thus $R_{temp}(u)$ high), the optimal choice for w is $w = u$, so that u is now effectively regularized by R_{spat}

Infimal Convolution regularization

Now we have a new regularizer

$$R(u) = \min_{w} R_{temp}(u - w) + R_{spat}(w)$$

Interpretation:

- If the warp is imprecise (and thus $R_{temp}(u)$ high), the optimal choice for w is $w = u$, so that u is now effectively regularized by R_{spat}
- In the opposite case, where R_{spat} would lose too many details, the optimal choice for w is $w = 0$.

Infimal Convolution regularization

Now we have a new regularizer

$$R(u) = \min_w R_{temp}(u - w) + R_{spat}(w)$$

Interpretation:

- If the warp is imprecise (and thus $R_{temp}(u)$ high), the optimal choice for w is $w = u$, so that u is now effectively regularized by R_{spat}
- In the opposite case, where R_{spat} would lose too many details, the optimal choice for w is $w = 0$.
- w is high in regions with imprecise optical flow and low in regions where details can be gained from R_{temp}

Infimal Convolution regularization

Now we have a new regularizer

$$R(u) = \min_{w} R_{temp}(u - w) + R_{spat}(w)$$

Interpretation:

- If the warp is imprecise (and thus $R_{temp}(u)$ high), the optimal choice for w is $w = u$, so that u is now effectively regularized by R_{spat}
- In the opposite case, where R_{spat} would lose too many details, the optimal choice for w is $w = 0$.
- w is high in regions with imprecise optical flow and low in regions where details can be gained from R_{temp}

Infimal Convolution regularization

Now we have a new regularizer

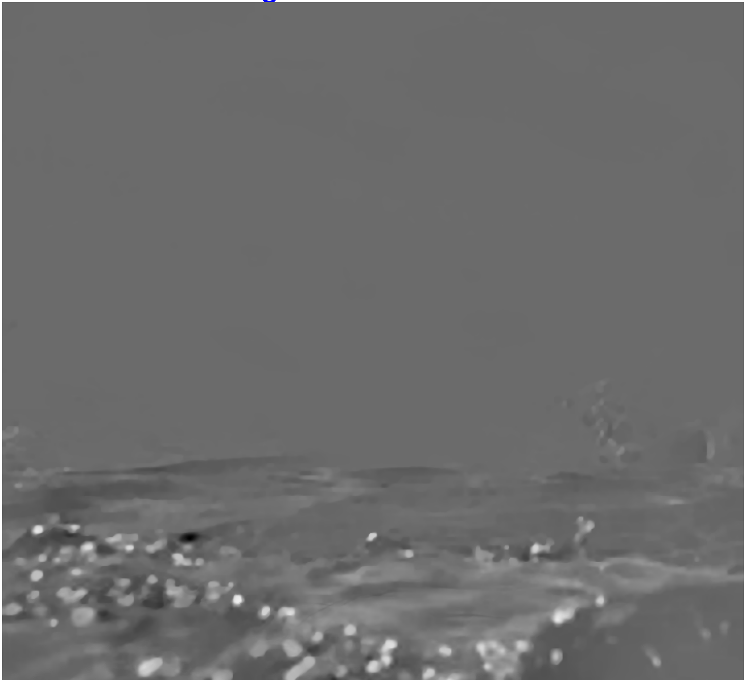
$$R(u) = \min_w R_{temp}(u - w) + R_{spat}(w)$$

Interpretation:

- If the warp is imprecise (and thus $R_{temp}(u)$ high), the optimal choice for w is $w = u$, so that u is now effectively regularized by R_{spat}
- In the opposite case, where R_{spat} would lose too many details, the optimal choice for w is $w = 0$.
- w is high in regions with imprecise optical flow and low in regions where details can be gained from R_{temp}

$\Rightarrow$ The infimal convolution adaptively balances both regularizers.

Infimal convolution regularization



Infimal convolution regularization - Making things more complicated

The infimal convolution defined on the last slide compares 'all spatial' regularization with 'all temporal' regularization.

Infimal convolution regularization - Making things more complicated

The infimal convolution defined on the last slide compares ’all spatial’ regularization with ’all temporal’ regularization.

Better (= Less error prone results) can be achieved by coupling to different mixings of temporal and spatial regularization.

Infimal convolution regularization - Making things more complicated

The infimal convolution defined on the last slide compares 'all spatial' regularization with 'all temporal' regularization.

Better (= Less error prone results) can be achieved by coupling to different mixings of temporal and spatial regularization.

$$R^1_\kappa(u) = \left\| \begin{pmatrix} \kappa W u \\ \nabla u \end{pmatrix} \right\|_{2,1}$$

$$R^2_\kappa(u) = \left\| \begin{pmatrix} W u \\ \kappa \nabla u \end{pmatrix} \right\|_{2,1}$$

for some mixing parameter $\kappa \in]0, 0.5[$

Infimal convolution regularization - Making things more complicated

Further improvements can be made if the optimal weighting between the operators W and ∇ is estimated on the automatically:

Infimal convolution regularization - Making things more complicated

Further improvements can be made if the optimal weighting between the operators W and ∇ is estimated on the automatically:

$$h = \frac{\|\mathcal{W} u_0\|_1}{\|\partial_x u_0\|_1 + \|\partial_y u_0\|_1}.$$

with the help of a bicubic zooming estimate u_0.

Infimal convolution regularization - Making things more complicated

Further improvements can be made if the optimal weighting between the operators W and ∇ is estimated on the automatically:

$$h = \frac{\|\mathcal{W}u_0\|_1}{\|\partial_x u_0\|_1 + \|\partial_y u_0\|_1}.$$

with the help of a bicubic zooming estimate u_0.
$\Rightarrow$ Incorporate this weighting:

$$R^1_\kappa(u) = || \begin{pmatrix} \frac{\kappa}{h} Wu \\ \nabla u \end{pmatrix} ||_{2,1}$$

$$R^2_\kappa(u) = || \begin{pmatrix} \frac{1}{h} Wu \\ \kappa \nabla u \end{pmatrix} ||_{2,1}$$

More considerations

There is (of course) a long list of further considerations

More considerations

There is (of course) a long list of further considerations

- Converting color images into YCbCr color spaces and computing the upsampling only on the Y channel.

More considerations

There is (of course) a long list of further considerations

- Converting color images into YCbCr color spaces and computing the upsampling only on the Y channel.
- Varying the blur kernel used before the downsampling (bicubic kernel, bilinear kernel, Lanczos kernel ...)

More considerations

There is (of course) a long list of further considerations

- Converting color images into YCbCr color spaces and computing the upsampling only on the Y channel.
- Varying the blur kernel used before the downsampling (bicubic kernel, bilinear kernel, Lanczos kernel ...)
- Alternating the directions of the warp computation to improve stability and reduce systematic errors

More considerations

There is (of course) a long list of further considerations

- Converting color images into YCbCr color spaces and computing the upsampling only on the Y channel.
- Varying the blur kernel used before the downsampling (bicubic kernel, bilinear kernel, Lanczos kernel ...)
- Alternating the directions of the warp computation to improve stability and reduce systematic errors

More considerations

There is (of course) a long list of further considerations

- Converting color images into YCbCr color spaces and computing the upsampling only on the Y channel.
- Varying the blur kernel used before the downsampling (bicubic kernel, bilinear kernel, Lanczos kernel ...)
- Alternating the directions of the warp computation to improve stability and reduce systematic errors

... But I think we will at this point, with some video demonstrations.

Video demo