CV Repetitorium - Test 2

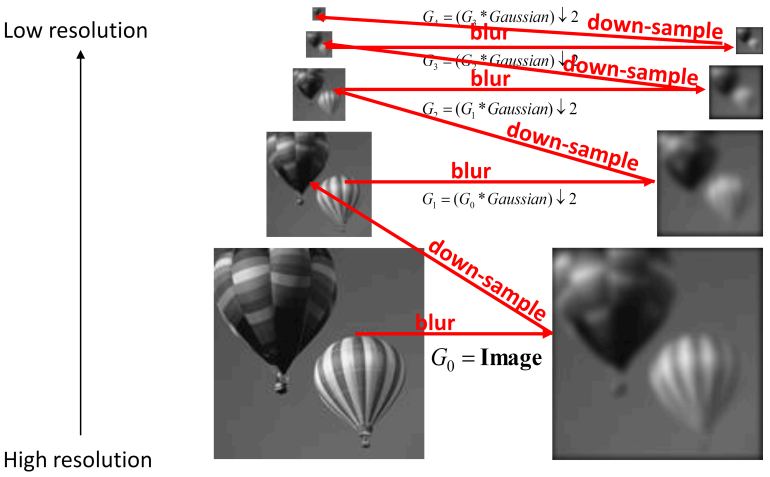
Multiscale Operations

(1)

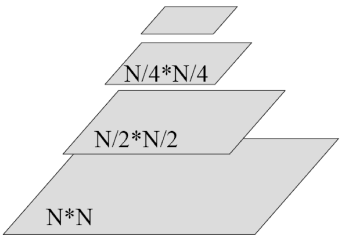
Multiscale Operations

- Assume that an image of the 1st level of a Gaussian pyramid consists of 24336 pixels (156x156), which corresponds to the original resolution of the image. On the 3rd level of the pyramid, the image still has **1521** pixels.

- To avoid aliasing artefacts when constructing image pyramids, the image must be **smoothed** before the size reduction.

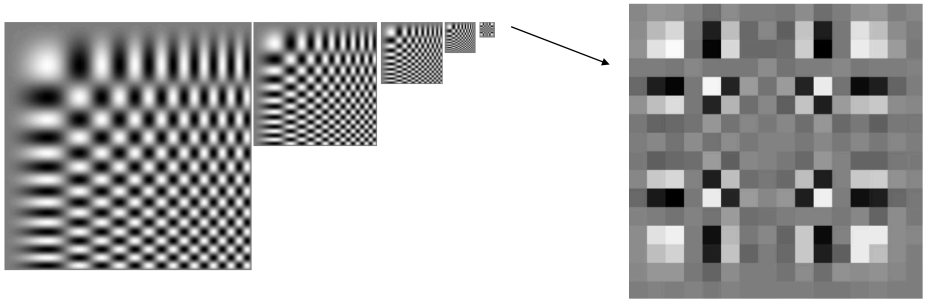


- Geometric series



$$N^2 + \frac{1}{4}N^2 + \frac{1}{16}N^2 + \cdots = 1\frac{1}{3}N^2$$

- Constructing a pyramid by taking every second pixel leads to layers that badly misrepresent the top layer



(2)

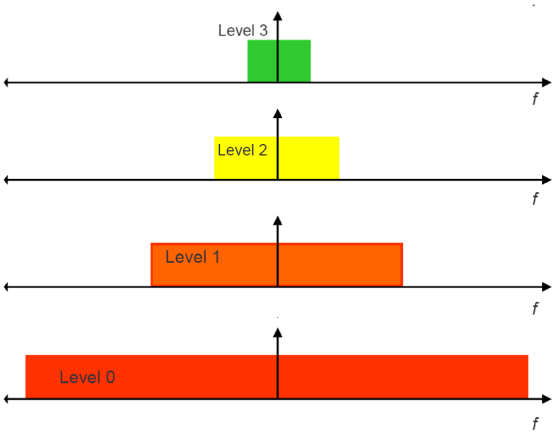
Multiscale Operations

In a Gaussian pyramid, the smaller images contain fewer high frequency components than the larger ones.	☒ True	☐ False
Die Laplacepyramide kann durch die Differenz von gaußgefilterten Bildern approximiert werden.	☒ True	☐ False
The Laplacian pyramid can be approximated by the difference of Gaussian-filtered images.	☒ True	☐ False
With bilinear interpolation, the new value is calculated from the 4 neighbouring values after an image reduction.	☒ True	☐ False

Begründungen:

Gaussian Pyramid Frequency Composition

- Lowpass filter



Difference of Gaussians (DoG)

- Laplacian of Gaussian can be approximated by the difference between two different Gaussians

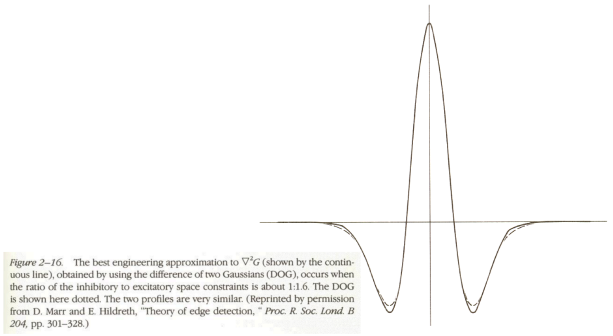
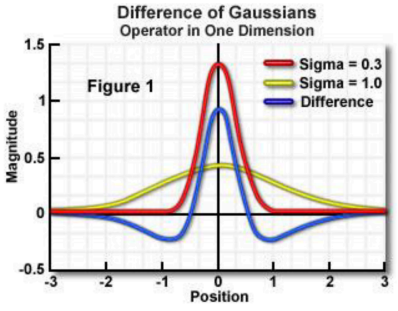
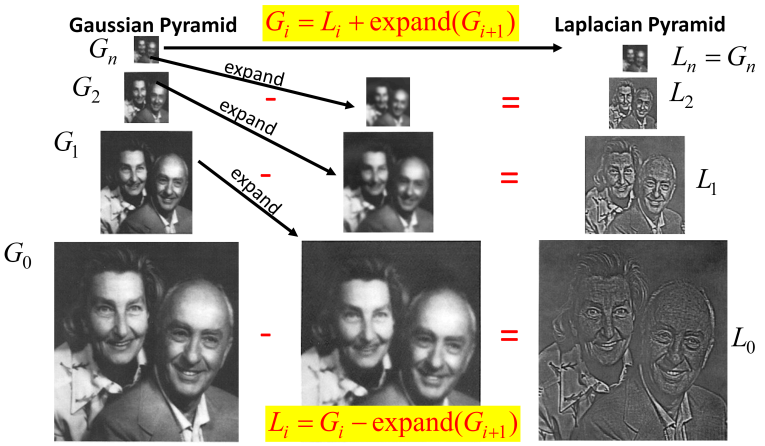


Figure 2–16. The best engineering approximation to $\nabla^2 G$ (shown by the continuous line), obtained by using the difference of two Gaussians (DOG), occurs when the ratio of the inhibitory to excitatory space constraints is about 1:1.6. The DOG is shown here dotted. The two profiles are very similar. (Reprinted by permission from D. Marr and E. Hildreth, "Theory of edge detection," *Proc. R. Soc. Lond. B 204*, pp. 301–328.)

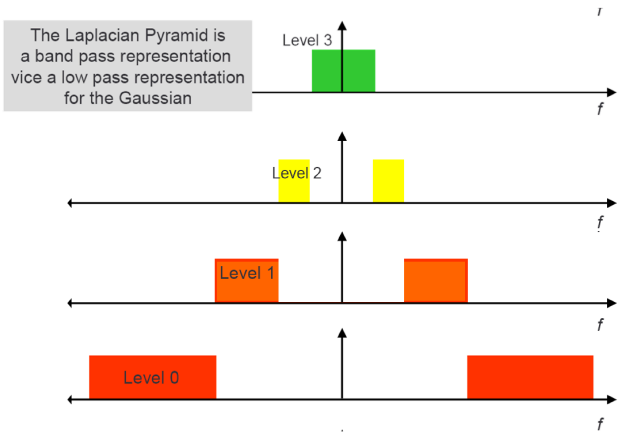


The Laplacian Pyramid

http://persci.mit.edu/pub_pdfs/RCA84.pdf

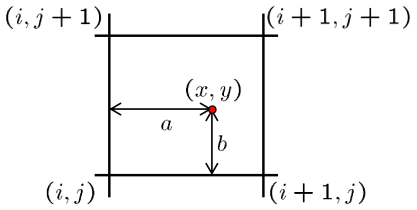


Laplacian Pyramid Frequency Composition



Bilinear interpolation

- A common method for resampling images
- If Area A != 1 -> Normalize by dividing by A



$$
\begin{aligned}
F(x,y) = \quad & (1-a)(1-b) && F(i,j) \\
& +a(1-b) && F(i+1,j) \\
& +ab && F(i+1,j+1) \\
& +(1-a)b && F(i,j+1)
\end{aligned}
$$

Interest Points

(1)

Bildmerkmale - Interest Points

Given a 5x5 image section to which the Moravec corner detector is to be applied. Calculate the changes in the intensities E for the point marked with an asterisk * (3,3) and the 4 displacements (1,0), (1,1), (0,1) and (-1,1). Use a window size of 3x3 and the sum of squared differences (SSD). Furthermore, determine the 'interest value' from the changes in intensity.

y					
1	70	60	70	60	60
2	80	80	90	80	80
3	80	90	**100***	100	100
4	80	90	100	100	100
5	70	80	100	100	110
x	1	2	3	4	5

$E(1,0)$= 400

$E(1,1)$= 1200

$E(0,1)$= 700

$E(-1,1)$= 1400

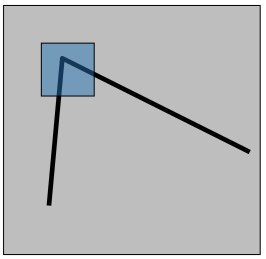
Interest value: 400

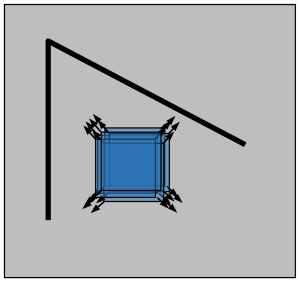
***E*(1,0):** SSD der Pixelwerte von ☐ und ☐ :
$(80-90)^2+(90-80)^2+(90-100)^2+(90-100)^2 = 400$
Interest value: Minimum der 4 Werte

Begründung

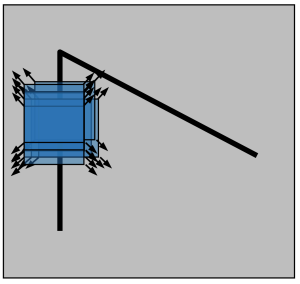
The Basic Idea

- We should easily localize the point by looking through a **small window**
- Shifting a window in **any direction** should give *a* **large change** in intensity

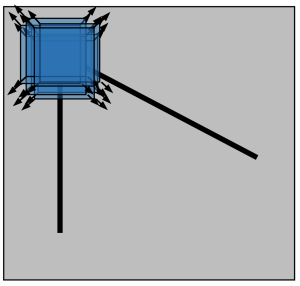




"flat" region:
no change as shift window in all directions



"edge":
no change as shift window **along the edge** direction



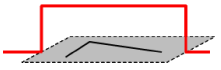
"corner":
significant change as shift window in **all directions**

- Change of intensity for the shift [u,v]:

$$E(u,v) = \sum_{x,y} w(x,y)\left[I(x+u,y+v) - I(x,y)\right]^2$$

Window function | Shifted intensity | Intensity

Window function $w(x,y)$ =



1 in window, 0 outside

Four shifts: (u,v) = (1,0), (1,1), (0,1), (-1, 1)
Look for local maxima in min{E}

(2)

Bildmerkmale - Interest Points

- Why is an orientation, the ‘dominant’ gradient direction of the image region, assigned to each interest point in SIFT?

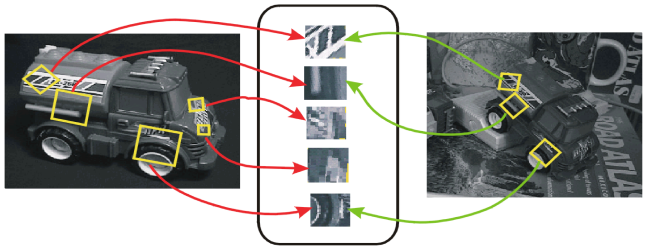
-> **Invariance to rotation**

- With SIFT, a feature vector is calculated using gradient histograms in 4x4 windows. A gradient histogram has 8 bins and the feature vector therefore has a total of **128** elements.

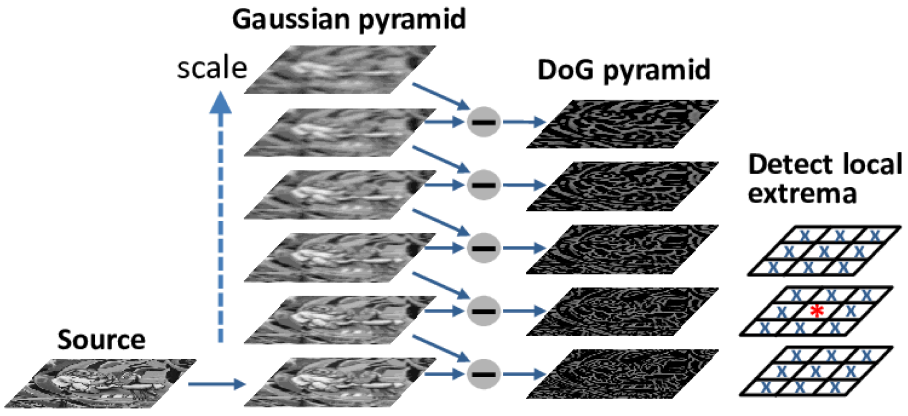
Begründung

SIFT (Scale Invariant Feature Transform)

- SIFT is an carefully designed procedure with empirically determined parameters for invariant and distinctive features
- Image content is transformed into local feature coordinates that are invariant to translation, rotation, scale, and other imaging parameters

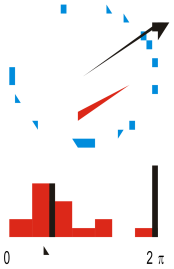


SIFT – Detecting Keypoints



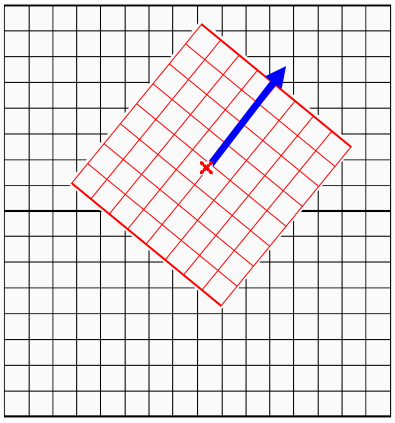
3. Finding Keypoints – Orientation

- Create histogram of local gradient directions computed at selected scale
- Assign canonical orientation at peak of smoothed histogram
- Each key specifies stable 2D coordinates (x, y, scale, orientation)



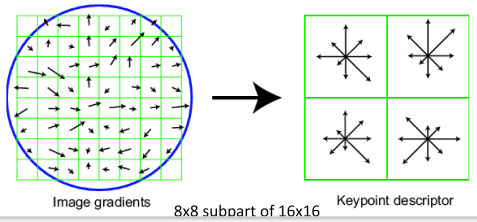
Finding Keypoints – Orientation

- Assign dominant orientation as the orientation of the keypoint



Stage 4: Keypoint Descriptor

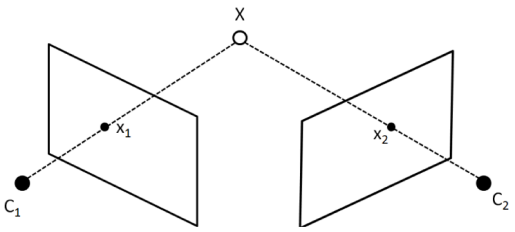
- **Image gradients** are sampled over 16x16 array of locations in scale space
- 16 4x4 windows with 8 orientations in each window
- Create array of **orientation histograms** for each window, amount added to each bin depends on the magnitude of the gradient and the distance from the interest point.
- 8 orientations x 4x4 histogram array = **128 dimensions**



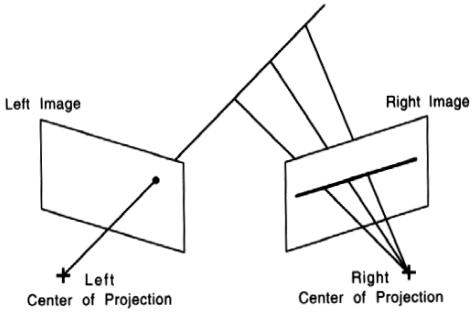
Stereo and Motion

kurzer Recap was das ist:

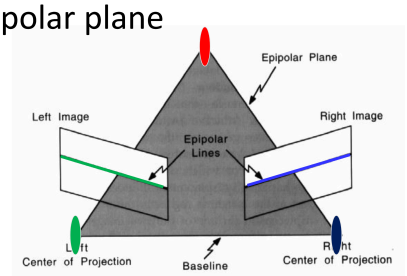
In the following sketch of a stereo system, a scene point X is projected onto the points x1 and x2 of the two cameras with the two focal points C1 and C2. Roughly draw the epipoles e1 and e2 as well as the two epipolar lines g1 and g2 in the sketch.



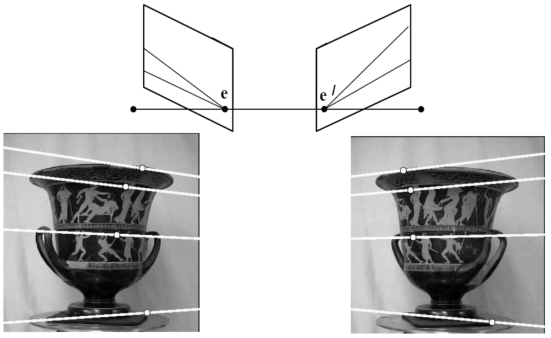
- **Epipolar Constraint**: Each point of the left image can lie only on a specific line in the right image: the Epipolar Line



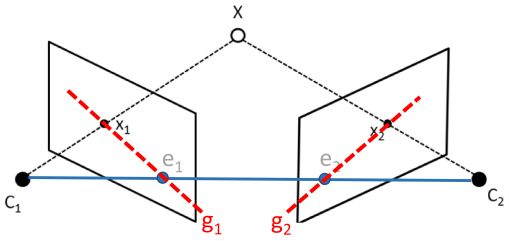
- Epipolar geometry is a consequence of the coplanarity of the camera centers and scene point
- The camera centers, corresponding points and scene point lie in a single plane, known as the epipolar plane



- Note, epipolar lines are in general not parallel



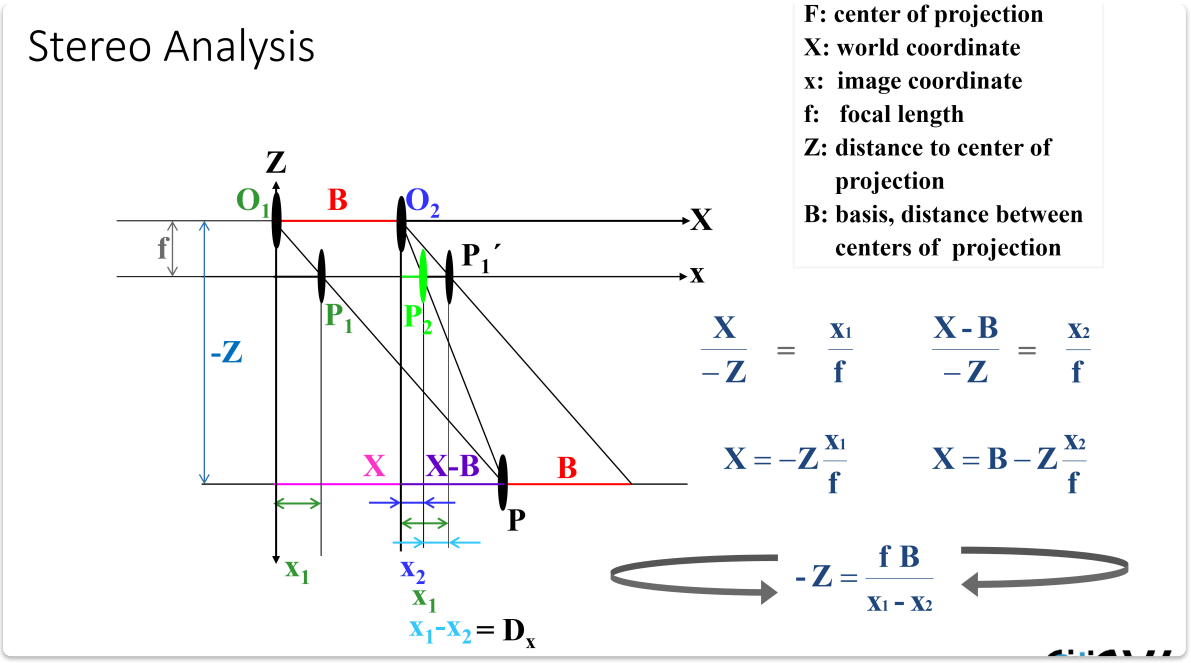
In the following sketch of a stereo system, a scene point X is projected onto the points x1 and x2 of the two cameras with the two focal points C1 and C2. Roughly draw the epipoles e1 and e2 as well as the two epipolar lines g1 and g2 in the sketch.



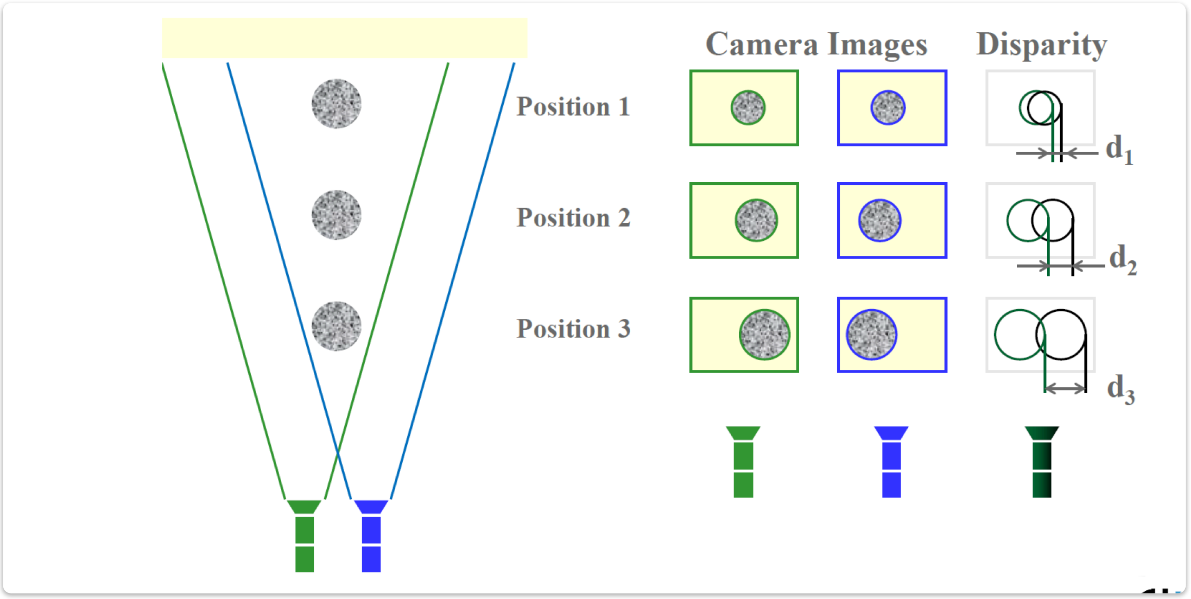
(1)

- A scene is recorded with a stereo setup. The distance between the two cameras with a focal length of 450 pixels is 8cm. A disparity of 10 pixels is determined for one pixel. How far away is the corresponding scene point?
  (450/10) * 8cm = 360cm
  - What is the disparity of a scene point that is twice as far away?
    5

Begründung



Wichtige Formel



(2)

- In contrast to feature-based matching, **area-based matching** calculates the depth values of **all** pixels in the image.

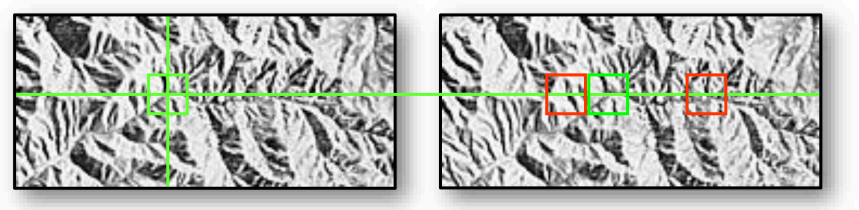
Begründung

- Area Based
  - Comparison of **intensity values** in the left and right image
  - Correspondence due to the **similarity** between the **intensity values**
  - Correspondence **for each pixel**

Left Stereo Image Right Stereo Image

- Feature Based
  - Comparison of **features** in left and right image
  - Correspondence on the basis of **selected characteristics** of features (edge orientation, edge length, gradient, etc.)
  - Correspondence only for **selected pixels**
  - **more accurate** because of sub-pixel positioning of features

- Finding pixel-to-pixel correspondences
  - For each pixel in the left image, search for the most similar pixel in the right image
  - Using neighborhood windows



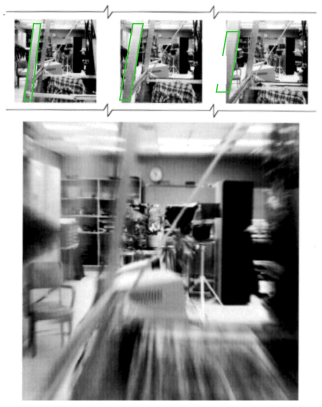
(3)

- The process of obtaining three-dimensional information about objects or an entire scene by analysing a temporal sequence of images is called **Structure-from-Motion**

Begründung

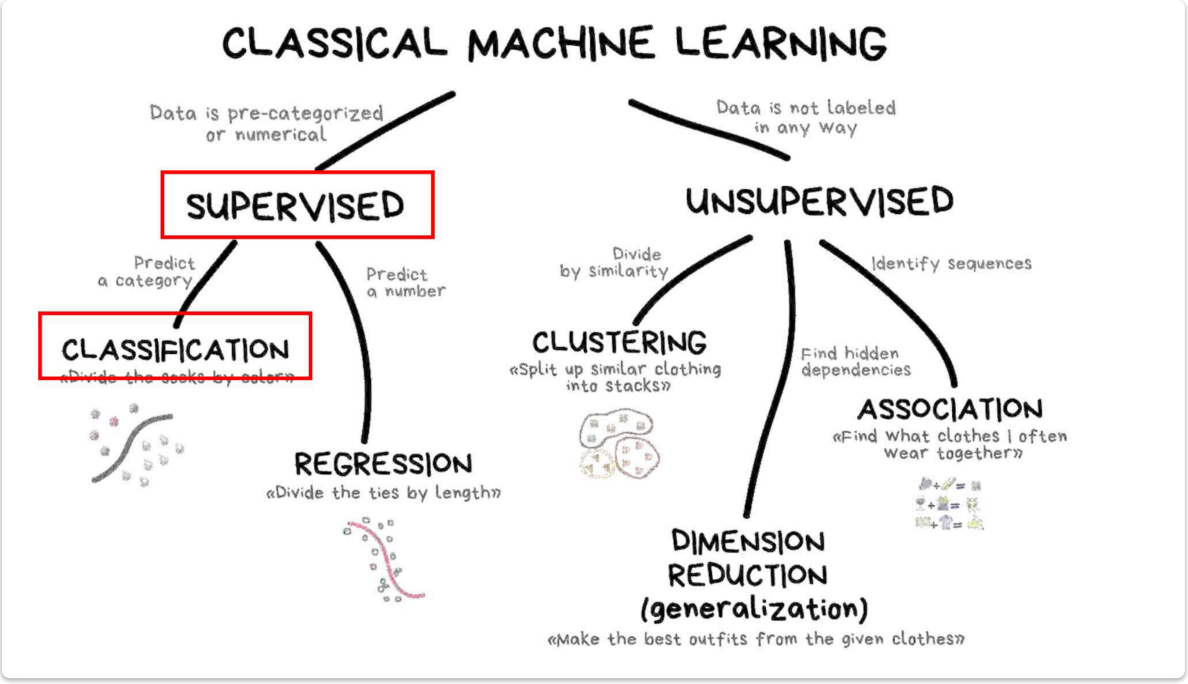
Structure from Motion

- Motion of an observer relative to the environment:
  - Information about **movement** of viewer
  - **Depth information** of the environment (cf. stereo)
  - Motion parallax = **Motion disparity**
- Problem: direction and amount of camera movement
  - Motion field **estimation**
  - Motion field **analysis**
  - Shift of viewpoint => motion of objects



Machine Learning & Deep Learning

recap

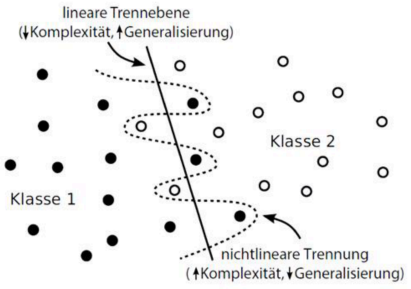


(1)

- An undesirable effect of machine learning is that a classifier only achieves a good result on the training data, but not on the test data. This effect is called **Overfitting**
- A classifier with a lower bias usually has a higher **variance** and vice versa.
- What is the name of a function that decides whether an artificial neuron transmits a signal or not? **Activation function**

Stoff dazu

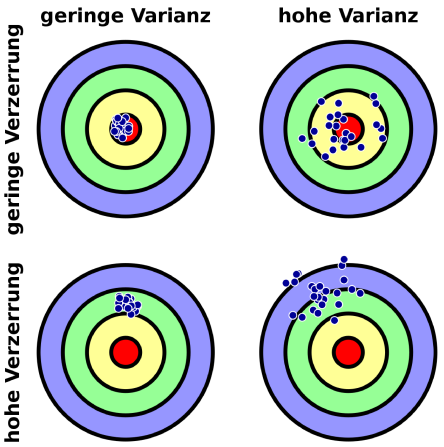
Generalization in Deep Learning

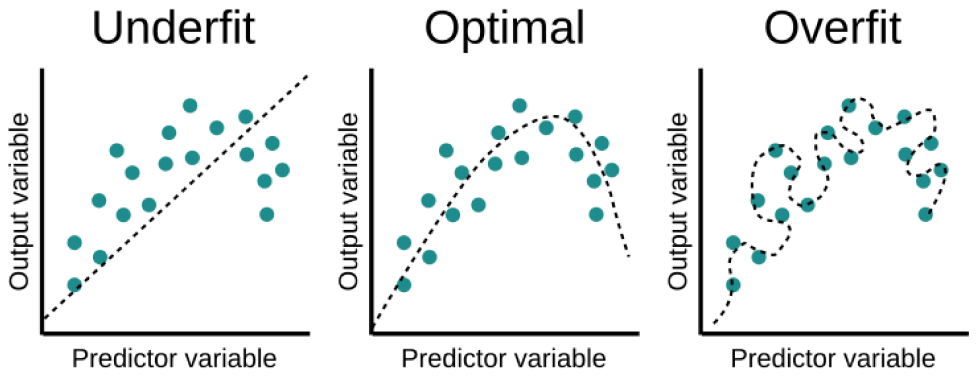


- **Bias**: how much the **average model** over all training sets **differ** from the **true model**?
- **Variance**: how much the **models** estimated from different training sets **differ from each other?**
- **Underfitting**: model is **too "simple"** to represent all the relevant class haracteristics
  - High bias and low variance
  - High training error and high test error
- **Overfitting**: model is **too "complex"** and fits **irrelevant characteristics** (noise) in the data
  - Low bias and high variance
  - Low training error and high test error

Bias-Variance Tradeoff

- Desired: low bias + low variance
- Eliminating bias
  - use local information
- Eliminating variance
  - average over multiple samples, use more global information

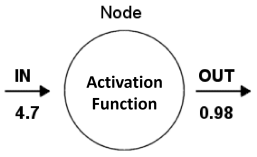




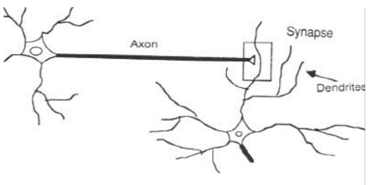
Generalization in Deep Learning

Node and Weight

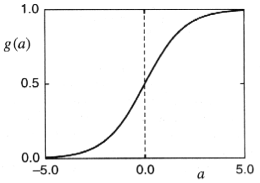
- Structure of a node:

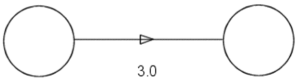


Synapse vs. Weight



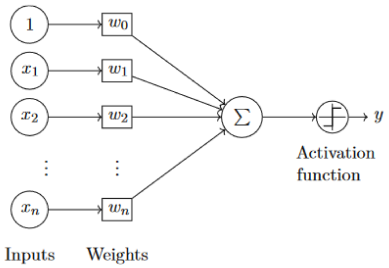
- Activation function
  - limits node output
  - **Nonlinearity**!





Perceptron

- Initial proposal of connectionist networks
- Rosenblatt, 50's and 60's
- Essentially a linear discriminant composed of nodes, weights



Weights

$$y = f(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x} + w_0)$$

Activation function

Bias

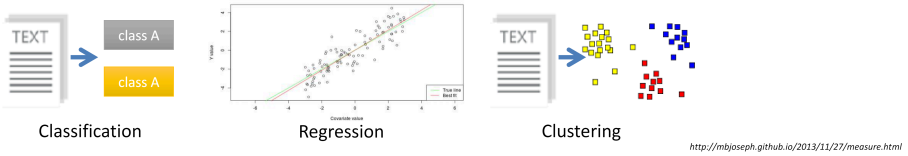
(2)

For supervised learning, the labels of the training examples must be known.	☒ True	☐ False
An auto-encoder has the task of finding an optimal higher-dimensional representation for a representation	☐ True	☒ False
In order to apply deep learning, features ('hand-crafted features') must be manually extracted from the images	☐ True	☒ False

Begründung und Stoff dazu

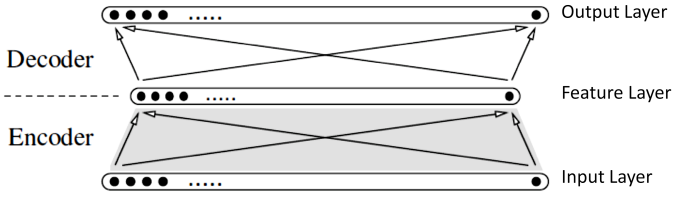
Types of Learning

- **Supervised**: Learning with a **labeled training** set
  - Example: email **classification** with already labeled emails
- **Unsupervised**: Discover **patterns in unlabeled** data
  - Example: **cluster** similar documents based on text
- **Reinforcement learning**: learn to **act** based on **feedback/reward**
  - Example: learn to play Go, reward: **win or lose**, ChatGPT



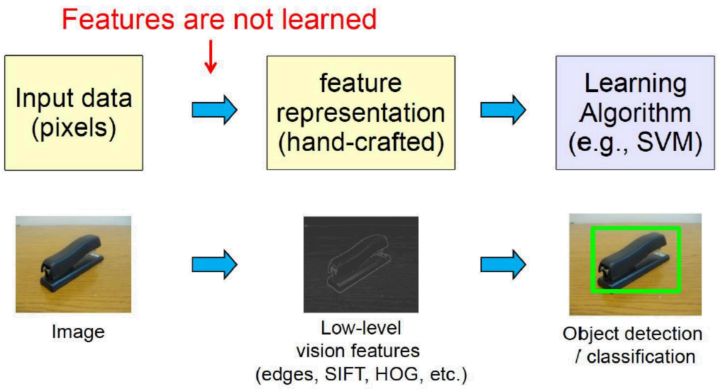
Auto-Encoder

- An auto-encoder is trained to reproduce the input (identity function)
  - Target == Input



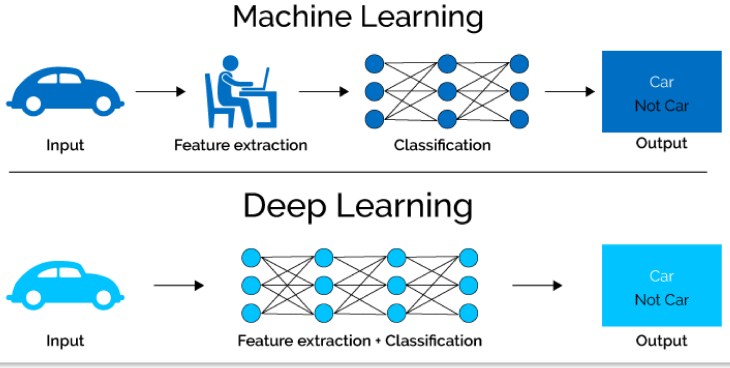
- Dim(Feature Layer) < Dim(Input Layer)
- **Image compression by learning an efficient encoding of the input**

Traditional Recognition Approach



Difference of Deep Learning to Classic Machine Learning

- Computes features
- Has simple and complex features
- Does not need human interaction



Computational Photography

(1)

- What is the size of the transformation matrix that can be used to perform an affine transformation between homogeneous image coordinates? **3x3**

Begründung

Parametric (Global) Warping

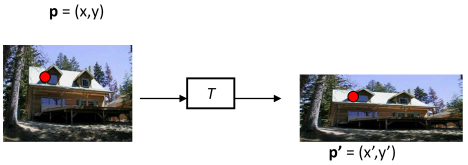
- Transformation T is a coordinate-changing machine:

  p' = *T*(p)

- What does it mean that *T* is global?
  - Is the same for any point p
  - can be described by just a few numbers (parameters)
- Let's represent *T* as a matrix:

  p' = **M**p

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \mathbf{M} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$



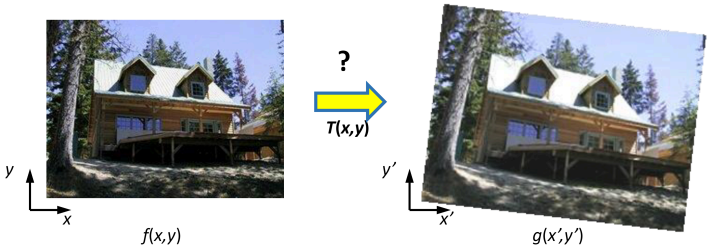
(2)

- How many corresponding image point pairs are required to determine any affine transformation between two images? **3**

Begründung

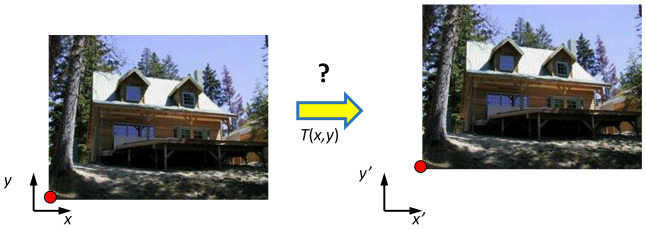
Recovering Transformations

- What if we know *f* and *g* and want to recover the transform T?
  - Willing to let user provide correspondences
  - How many do we need?



Translation: # Correspondences?

- How many correspondences needed for translation? 1
- How many Degrees of Freedom? 2
- What is the transformation matrix?

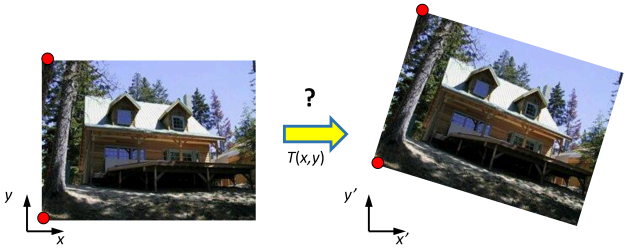


$$T = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix}$$

Euclidian/Rigid: # Correspondences?

- How many correspondences needed for translation + rotation? 2
- How many DOF? 3

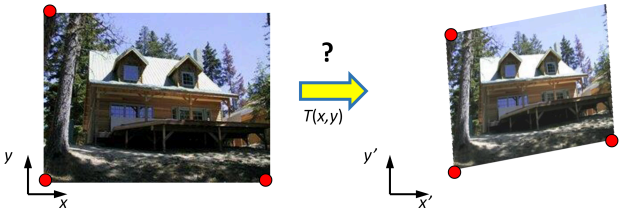
$$T = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & t_x \\ \sin(\theta) & \cos(\theta) & t_y \\ 0 & 0 & 1 \end{bmatrix}$$



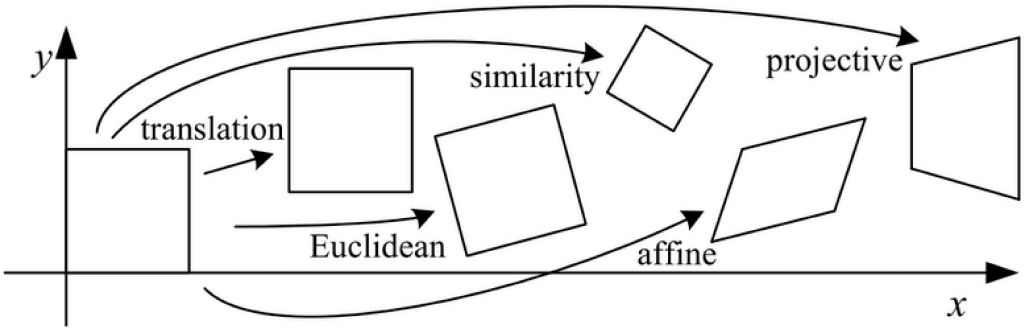
Affine: # Correspondences?

- How many correspondences needed for affine? 3
- How many DOF? 6

$$T = \begin{bmatrix} a & b & t_x \\ d & e & t_y \\ 0 & 0 & 1 \end{bmatrix}$$



Transformations



- Euclidean = rigid
- Similarity: equal scale factor in x and y

(3)

Computational Photography

The following corresponding points between two images are given:

$$p_1 = \begin{pmatrix}1\\1\end{pmatrix} p_1' = \begin{pmatrix}13\\13\end{pmatrix} \quad p_2 = \begin{pmatrix}5\\1\end{pmatrix} p_2' = \begin{pmatrix}5\\13\end{pmatrix} \quad p_3 = \begin{pmatrix}4\\2\end{pmatrix} p_3' = \begin{pmatrix}7\\11\end{pmatrix}$$

where pi is a point in the first image and p'i is the corresponding point in the second image. Determine the missing elements of the transformation matrix T that describes the image transformation from the first to the second image:

$$T = \begin{pmatrix} a & b & c \\ 0 & -2 & 15 \\ 0 & 0 & 1 \end{pmatrix}$$

Computational Photography

The following corresponding points between two images are given:

$$p_1 = \begin{pmatrix}1\\1\end{pmatrix} p_1' = \begin{pmatrix}13\\13\end{pmatrix} \quad p_2 = \begin{pmatrix}5\\1\end{pmatrix} p_2' = \begin{pmatrix}5\\13\end{pmatrix} \quad p_3 = \begin{pmatrix}4\\2\end{pmatrix} p_3' = \begin{pmatrix}7\\11\end{pmatrix}$$

where pi is a point in the first image and p'i is the corresponding point in the second image. Determine the missing elements of the transformation matrix T that describes the image transformation from the first to the second image:

$$T = \begin{pmatrix} a & b & c \\ 0 & -2 & 15 \\ 0 & 0 & 1 \end{pmatrix}$$

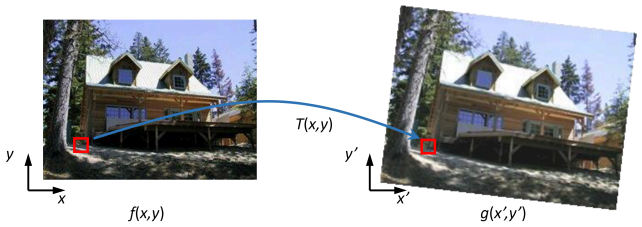
p´1 = T · p1 -> 1·a+1·b+1·c = 13
p´2 = T · p2 -> 5·a+1·b+1·c = 5
p´3 = T · p3 -> 4·a+2·b+1·c = 7

a=-2, b=0, c=15

Begründung

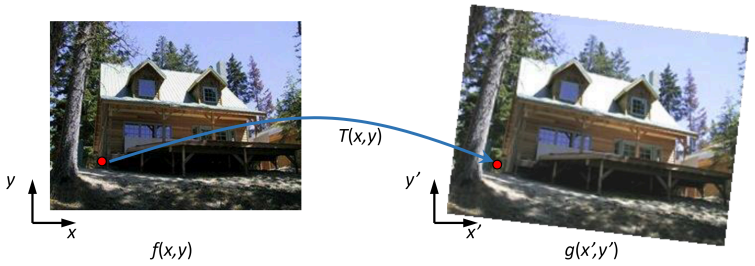
Image Warping

- Given a coordinate transform $(x',y') = T(x,y)$ and a source image $f(x,y)$, how do we compute a transformed image $g(x',y') = f(T(x,y))$?



Forward Warping

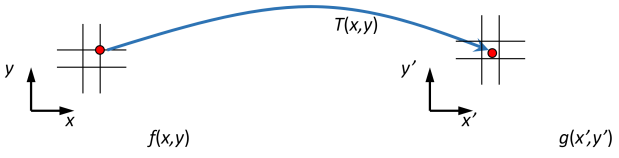
- Send each pixel $f(x,y)$ to its corresponding location
  $(x',y') = T(x,y)$ in the second image



Q: what if pixel lands "between" two pixels?

Forward Warping

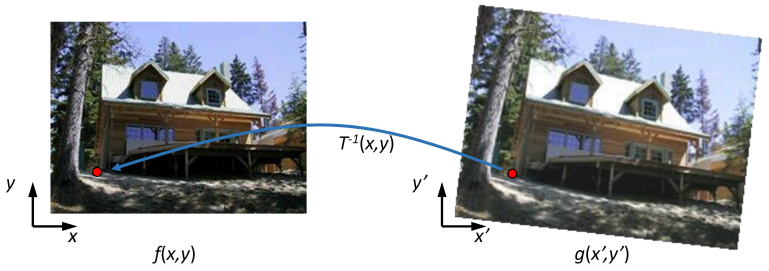
- Send each pixel $f(x,y)$ to its corresponding location
  $(x',y') = T(x,y)$ in the second image



Q: what if pixel lands "between" two pixels?
A: distribute color among neighboring pixels (x',y')
  – Known as "splatting"

Inverse Warping

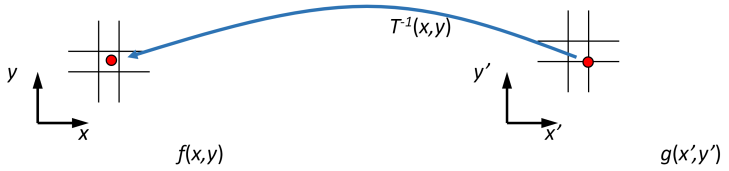
- Get each pixel $g(x',y')$ from its corresponding location
  $(x,y) = T^{-1}(x',y')$ in the first image



Q: what if pixel comes from "between" two pixels?

Inverse Warping

- Get each pixel $g(x',y')$ from its corresponding location
  $(x,y) = T^{-1}(x',y')$ in the first image

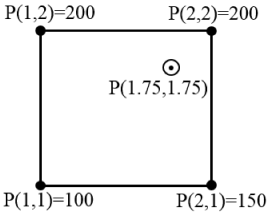


Q: what if pixel comes from "between" two pixels?

A: Interpolate color value from neighbors
- nearest neighbor, bilinear, Gaussian, bicubic

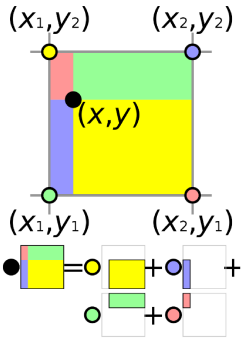
Computational Photography

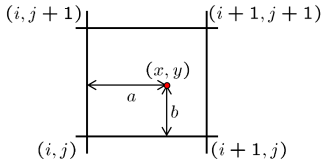
When resampling an image, the value of the point P(1.75,1.75) must be determined from the 4 neighbouring values P(1,1), P(1,2), P(2,1) and P(2,2) (see figure). Calculate the value by bilinear interpolation:



Bilinear Interpolation

- A common method for resampling images
- Assuming sum of weights = 1, Area = 1





$$
\begin{aligned}
F(x,y) = \quad & (1-a)(1-b) && F(i,j) \\
& +a(1-b) && F(i+1,j) \\
& +ab && F(i+1,j+1) \\
& +(1-a)b && F(i,j+1)
\end{aligned}
$$

(4)

Computational Photography

When resampling an image, the value of the point P(1.75,1.75) must be determined from the 4 neighbouring values P(1,1), P(1,2), P(2,1) and P(2,2) (see figure). Calculate the value by bilinear interpolation:

$0.25 \cdot 0.25 \cdot 100 +$ *-> P(1,1)*
$0.25 \cdot 0.75 \cdot 200 +$ *-> P(1,2)*
$0.75 \cdot 0.75 \cdot 200 +$ *-> P(2,2)*
$0.75 \cdot 0.25 \cdot 150$ *-> P(2,1)*

= 184.375