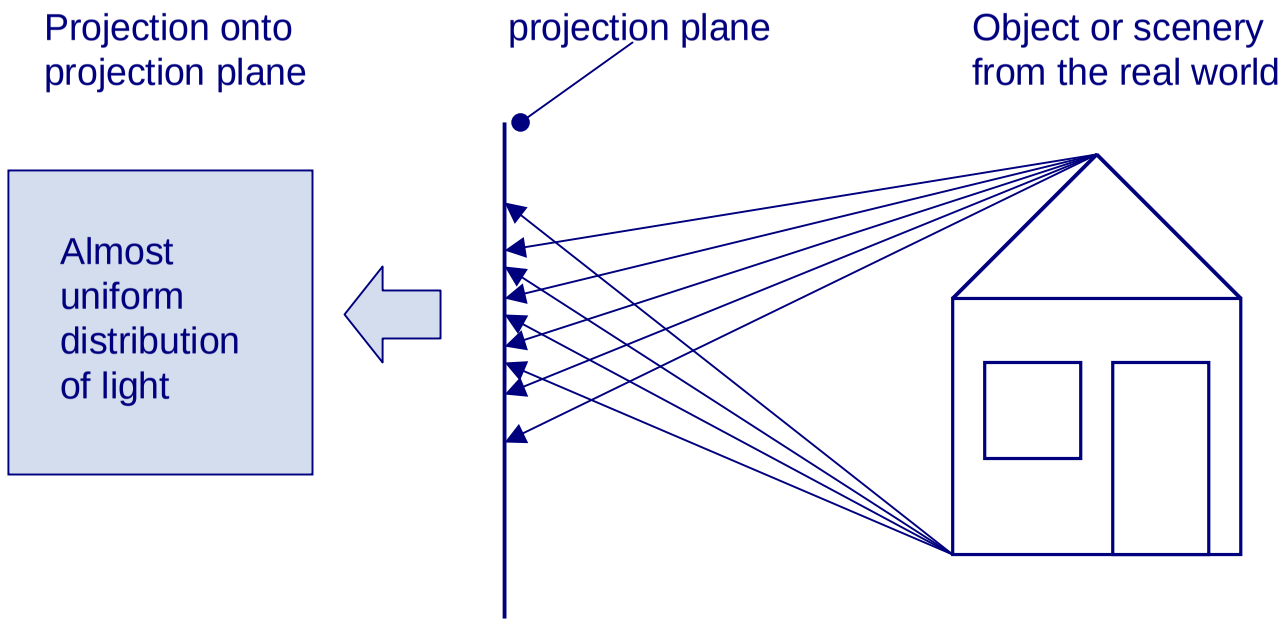
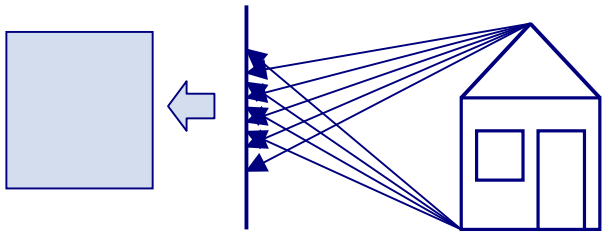
2.5.1 Brief historical introduction to animation

Capturing still images

Capturing and projecting still images preceeded the advent of animations. The idea of projecting images was not invented by anyone but is a natural phenomenon itself.



2.5.1 Brief historical introduction to animation



Every point on the surface of the object can be considered to be a tiny source of light, each one emitting rays of light in all directions uniformly.

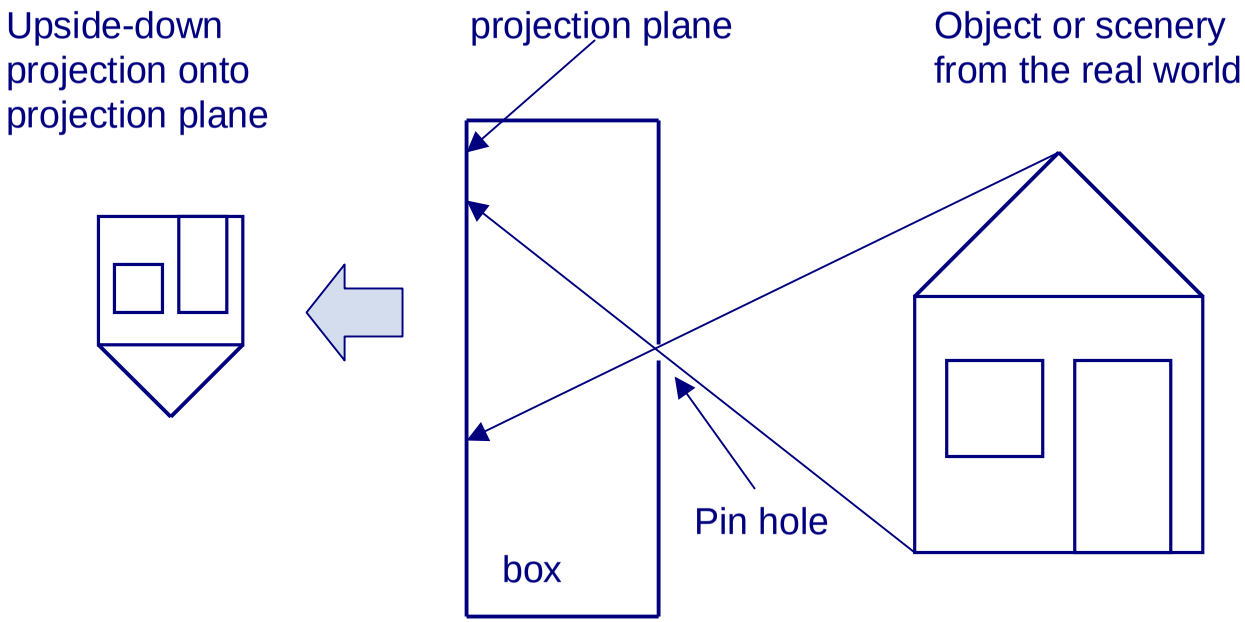
If no occlusion occurs it also emits almost equal amounts of light onto every other point of the projection plane. Obviously no sharp image can be expected, if all points on the surface of the object emit light onto all parts of the projection plane.

(Note: In 3D graphics the phenomenon that every part of an object is at the same time itself illuminated as well as source of light, is modelled by the so-called radiosity approaches.)

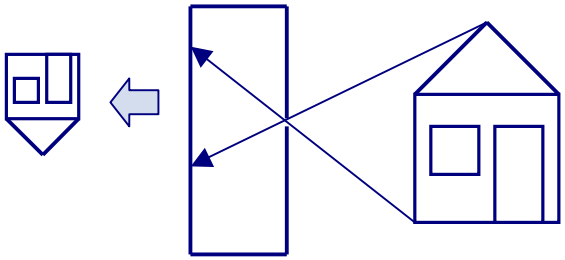
2.5.1 Brief historical introduction to animation

The pin hole camera or “camera obscura”

Surround the projection plane with a black box. Allow light only to pass through a tiny pin hole.



2.5.1 Brief historical introduction to animation



The black box will have blocked almost all rays. Only those points on the surface of the object are seen by the projection plane that are point-projected with the pin hole as the center of projection.

As is true for all point projections, the image is upside-down and mirrored at the same time (all axes change their direction).

The phenomenon was first discovered by the Chinese philosopher Mo-Ti (470 to 391 BC) and rediscovered and also mathematically explained by many others, such as Leonardo Da Vinci in 1490.

Before the invention of the photographic film, especially in the Middle Ages a painter would sit inside the black box and paint the projection onto the projection plane.

2.5.1 Brief historical introduction to animation

Picture (right) taken with a pinhole camera.
Pinhole size: 1/75 inch.
Film-to-pinhole distance: 4 1/2 inches.
Exposure: 2 seconds.
Subject in bright sunlight.

Smaller pinholes produce sharper images but require a longer exposure time.

Image by Eastman Kodak Company (C)

2.5.1 Brief historical introduction to animation

Projection of still images or the “laterna magica”

The so-called laterna magica first appeared in the 17th century. Again, no explicit inventor is known. It was used to project images rather than to capture them. The technique was very similar to that of the well-known slide projector.

Note that there is a smoke pipe on the device shown on the right since electric light had not yet been invented. However, lenses had already been invented in the 17th century so that a sharp projection could be produced.

The laterna magica was used to project images drawn on glass. Used mainly as an attraction at fun fairs, its first application for educational purpose occurred in the 19th century.

2.5.1 Brief historical introduction to animation

Early moving images

The first device to produce a moving picture appeared in the 19th century and was called a “phenakistiskop” also known as the “wheel of life”.

This idea is to draw a small number of steps of an animation on a disc. For each picture on the disc, there is a neigboring slot.

While standing in front of a mirror the viewer can see a single image through the slot if they are facing the opposite side of the disc. If the disc is spinning around its middle axis at a specific speed, the eye can not follow a single image (also due to the occlusion caused by the slots). Thus an animation is perceived.

The phenakistiskop was sold mainly as a toy. Later, more advanced versions using glass, lenses and mirrors were used in combination with the laterna magica to project moving images.

2.5.1 Brief historical introduction to animation

The invention of cinema

Cinema as a series of successive photos was invented by the Lumière brothers and by Thomas A. Edison in the late 19th century.

While Edison's camera was stationary and cumbersome, Lumiére's camera was portable and served as camera and projector at the same time.

Edison and Lumière each had a factory which produced both the devices and the films. However, the content of the films was of minor importance. Performing any kind of animation was enough for the audience.

2.5.2 Basic animation techniques

Conventional animation

- Write or choose a story
- A so-called story board is laid out
- Record the sound track (if any)
- Most important key frames are drawn by skilled artists
- The background and other moving layers (if any) have to be drawn
- All missing frames in between the key frames are drawn (= in betweening)
- An early trial film, the pencil test, is made to ensure a smooth animation
- All layers are put together and color is added to the animated characters

Obviously, the process is time-consuming and expensive because of the large number of images:

20 minutes of film * 25 frames/second = 30.000 frames

2.5.2 Basic animation techniques

Computer-assisted animation

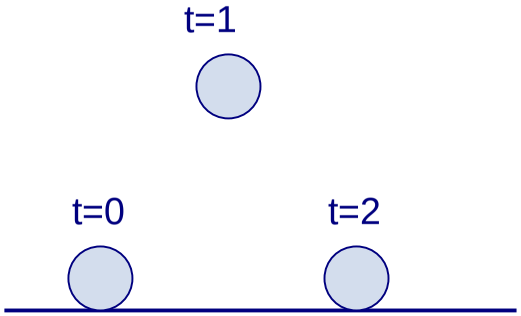
The idea is to digitize the early pencil drawings and filter the images to remove small dots or mistakes.

Now the painting process can be replaced by simple flood fills and the pencil test can be simulated by switching between different frame buffers.

2.5.2 Basic animation techniques

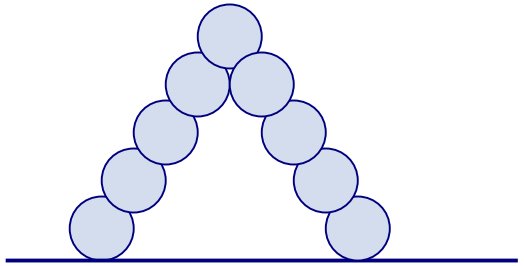
Digitally produced animations

Speed up the process even more by letting the computer do the in-betweening



Three stages of the jumping ball are defined as key frames.

The positions in between are calculated by linear interpolation (also called lerping = Linear intERPolation)



2.5.2 Basic animation techniques

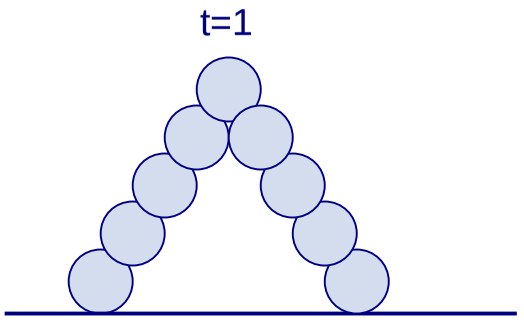
Animation by linear interpolation

$$location_{t+\alpha} = (1-\alpha) \bullet location_t + \alpha \bullet location_{t+1}$$

$$0 \le \alpha \le 1$$

Interpolation can also be applied to other parameters such as angles. Linear interpolation is simple; however, the result will not always be satisfying. This is especially true for the example of the flying ball.

The ball will change its direction in t=1 instantaneously if linear interpolation is used.

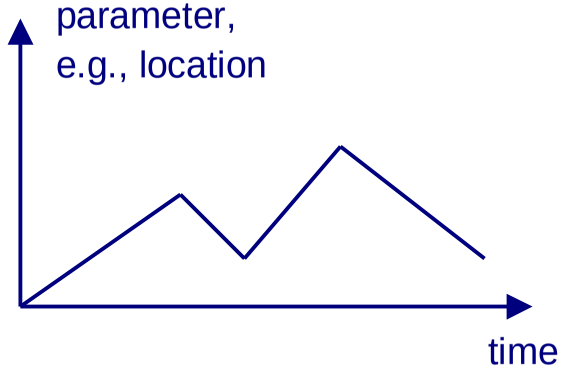


2.5.2 Basic animation techniques

Animation by linear interpolation

$$location_{t+\alpha} = (1-\alpha) \bullet location_t + \alpha \bullet location_{t+1}$$

$$0 \le \alpha \le 1$$



Linear interpolation means that the function is steady but the first derivation is not. If the parameter is mapped to a location, the object will not jump. But in order to obtain smooth movements, splines of a higher order have to be used.

Smoothness of the first derivation (which is the speed) is usually sufficient, as humans are not sensitive to higher-order derivations.

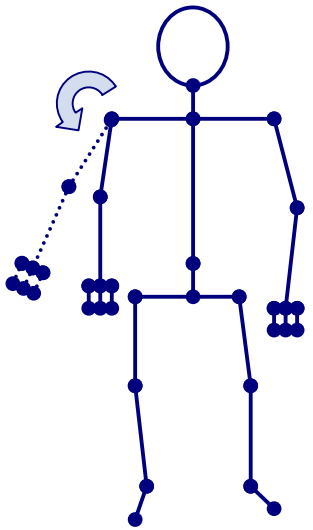
2.5.3 Articulated structures

Articulated structures

Articulated structures, or skeletons, are one technique to simplify animations. Using articulated structures which originate from the field of robotics is an essential first step in character animation and they are the basics for applying (inverse) kinematics.

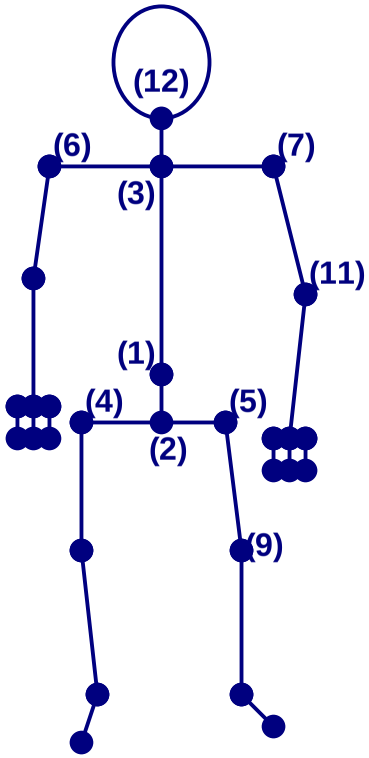
The skeleton is a two- or three-dimensional collection of connected, rigid segments. Only the joints themselfes have to be stored. They are connected in a tree-like structure. Two joints may implicitly be linked by a bone if there is a direct parent-child relationship (next page).

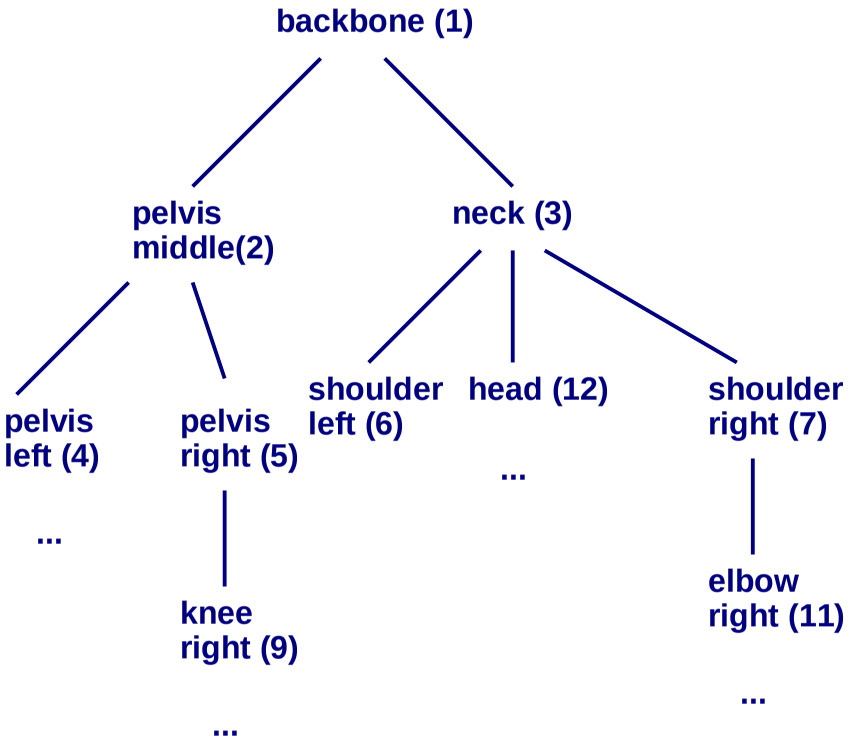
Advantage over an animation using no articulated structures: If a joint is rotated, all of its children have to follow the manipulation implicitly.



2.5.3 Articulated structures

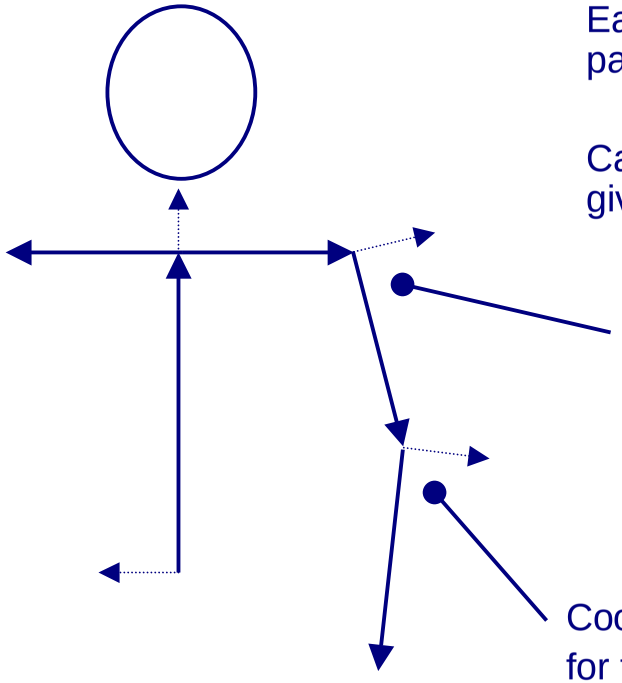
Articulated structures (example of a tree hierarchy)





2.5.3 Articulated structures

Articulated structures (representation of joints and bones)



Each bone is implicitly defined between a joint and its parent joint.

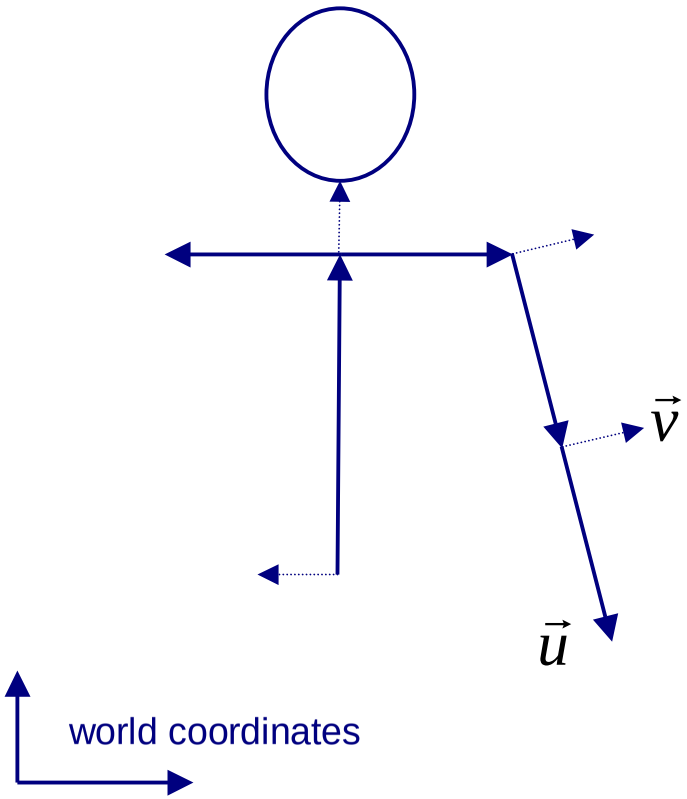
Calculating the orthogonal vector to each bone will give us a coordinate system for each joint.

Coordinate system for the shoulder

Coordinate system for the elbow

2.5.3 Articulated structures

Articulated structures (representation of joints and bones)



Caution: The coordinate systems in the figure are displayed in the world coordinate system. Note that each of them relates only to its parent coordinates!

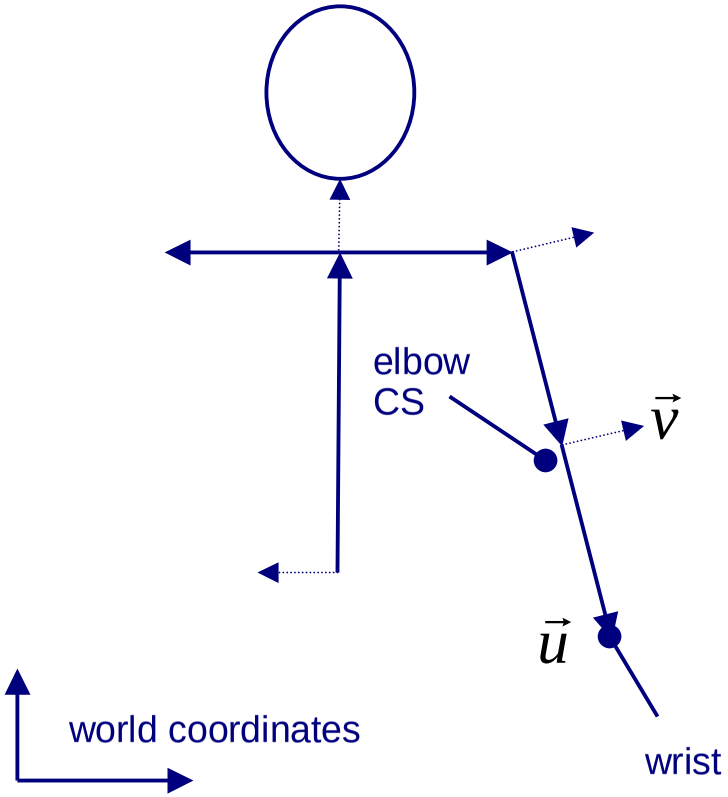
In this example the forearm is not rotated relative to the upper arm.

$$\begin{pmatrix} u_1' & v_1' \\ u_2' & v_2' \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

No rotation is equal to the unit matrix. u‘ and v‘ are equal to u and v resp., however, their lengths are normalized to 1 (which means: we want to rotate but we don‘t have to scale).

2.5.3 Articulated structures

Articulated structures (representation of joints and bones)



How are the acual world coodinates calculated?

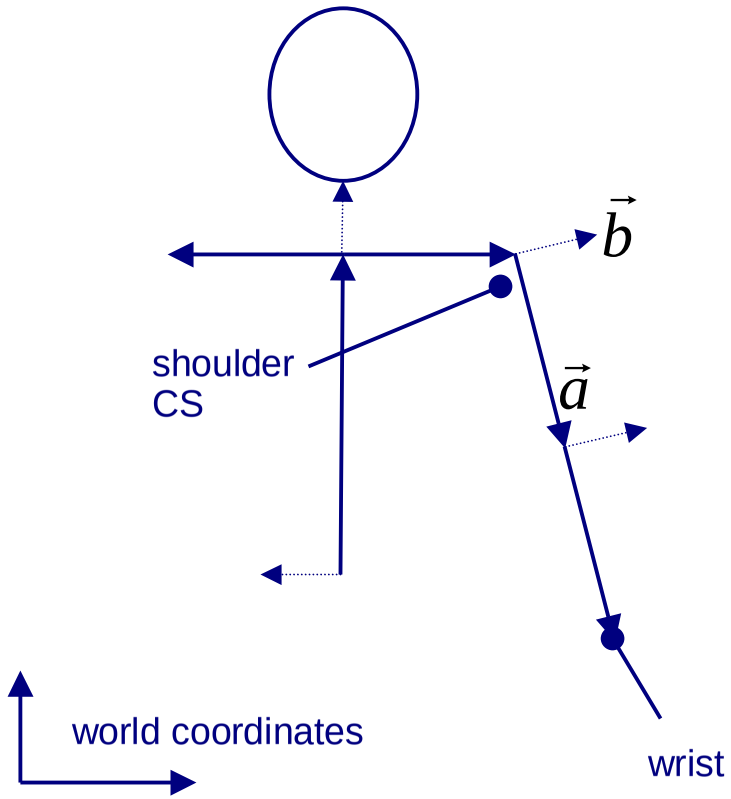
Example: Where is my wrist?

The wrist is located at the origin of its own CS (not shown in the figure) so we don‘t have to rotate it. So first of all it is defined in the elbow CS u, v. Since u, v also defined no rotation (see prev. page), we only have to add vector u. Why? Because the origin of the wrist CS is at the end of u. So far we only get:

$$\begin{pmatrix} u_1 \\ u_2 \end{pmatrix}$$

2.5.3 Articulated structures

Articulated structures (representation of joints and bones)



Next, the location of the wrist has to be calculated relative to the shoulder coordinate system

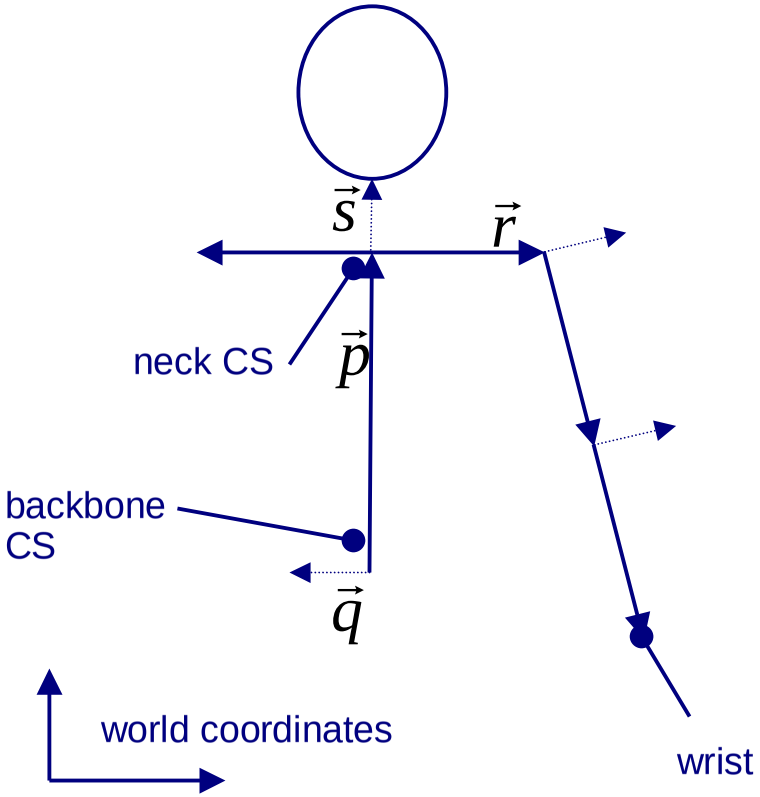
$$\begin{pmatrix} a_1' & b_1' \\ a_2' & b_2' \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \end{pmatrix}$$

Note that the center of the elbow CS is at the end of vector a of the shoulder CS. Thus we have to add vector a.

$$\begin{pmatrix} a_1' & b_1' \\ a_2' & b_2' \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \end{pmatrix} + \begin{pmatrix} a_1 \\ a_2 \end{pmatrix}$$

2.5.3 Articulated structures

Articulated structures (representation of joints and bones)



Now the location of the wrist has to be calculated relative to the neck coordinate system

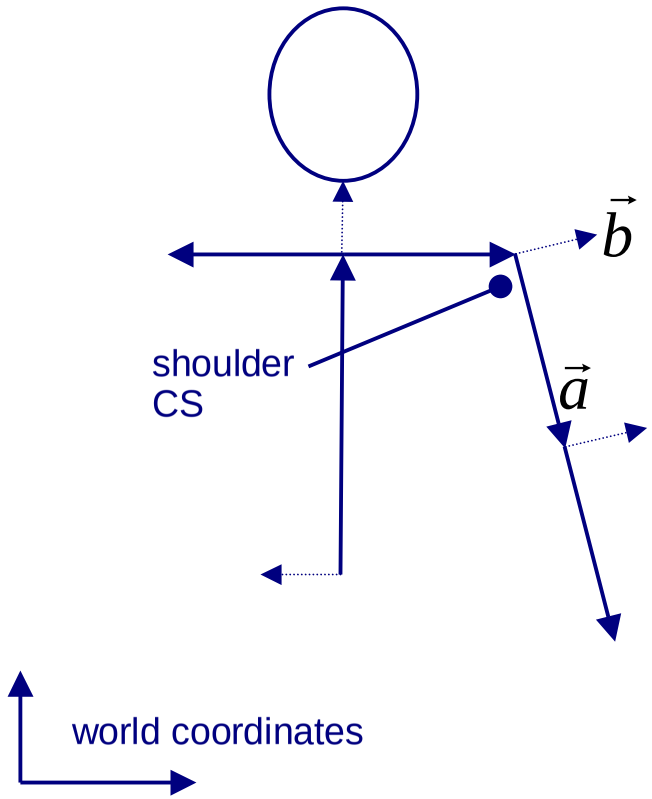
$$\begin{pmatrix} a_1' & b_1' \\ a_2' & b_2' \end{pmatrix}\begin{pmatrix} u_1 \\ u_2 \end{pmatrix}+\begin{pmatrix} a_1 \\ a_2 \end{pmatrix}$$

$$\begin{pmatrix} r_1' & s_1' \\ r_2' & s_2' \end{pmatrix}\left(\begin{pmatrix} a_1' & b_1' \\ a_2' & b_2' \end{pmatrix}\begin{pmatrix} u_1 \\ u_2 \end{pmatrix}+\begin{pmatrix} a_1 \\ a_2 \end{pmatrix}\right)+\begin{pmatrix} r_1 \\ r_2 \end{pmatrix}$$

... and so on

2.5.3 Articulated structures

Kinematics and inverse kinematics



Rather than defining the CS explicitly, we would prefer to define it as a rotation using an angle alpha.

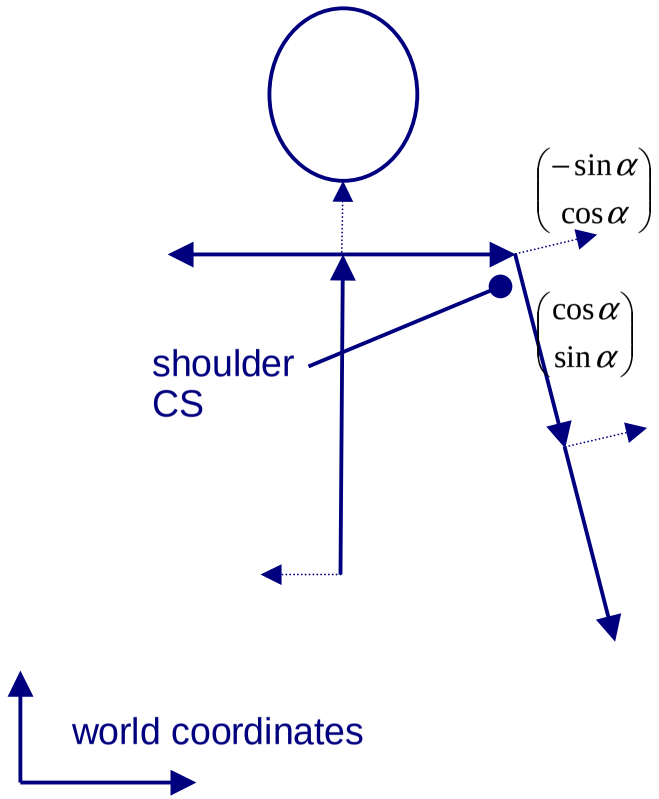
$$\begin{pmatrix} a_1 & b_1 \\ a_2 & b_2 \end{pmatrix} = \begin{pmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{pmatrix}$$

If a joint does not have the full degree of freedom, the parameter for the angle can be limited.

$$angle_{\min} < \alpha < angle_{\max}$$

2.5.3 Articulated structures

Kinematics and inverse kinematics



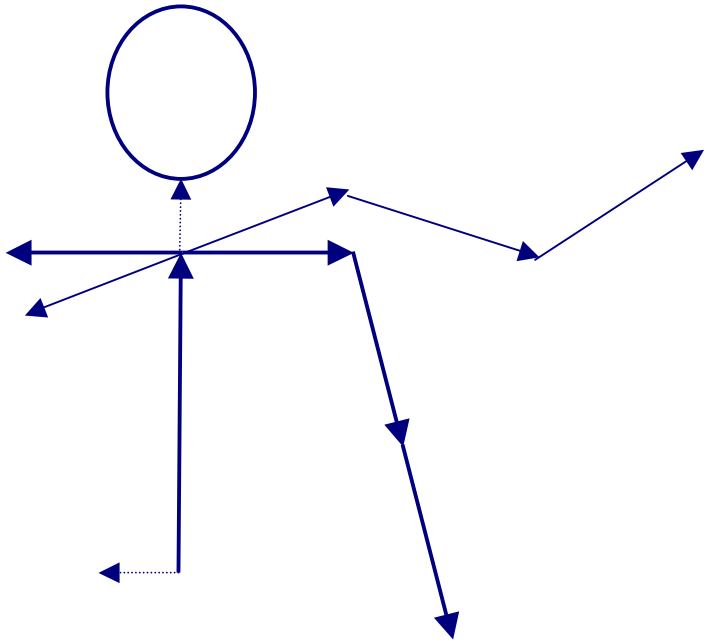
In forward kinematics, the angle of each joint is manipulated in every frame of the animation. Thus a smooth and realistic movement can be achieved.

Problem: How to manipulate the joints in order to reach a target, e. g., to grab something with a hand.

Increasing the number of joints will also increase the number of ways to animate a skeleton. However, not all of them correspond to a realistic motion.

2.5.3 Articulated structures

Kinematics and inverse kinematics



Solution: Use inverse kinematics.

A starting position and an ending position has to be found. In between all angles can be interpolated. If the starting- and ending-positions are credible, the same will usually be true for the stages in between.

How to find realistic starting- and ending-positions is still an open issue without any standard techniques. However, it is known that humans tend to keep their extremities in mostly low-energy positions and that they try to distribute their movements equally over all joints.

Often motion-capturing techniques are used to obtain realistic key positions of a character, e. g., in game development.

2.5.4 Image morphing

Basic idea of image morphing

In *image morphing*, also known as *image metamorphosis*, one object in a source image has to be transformed into another object in a target image.

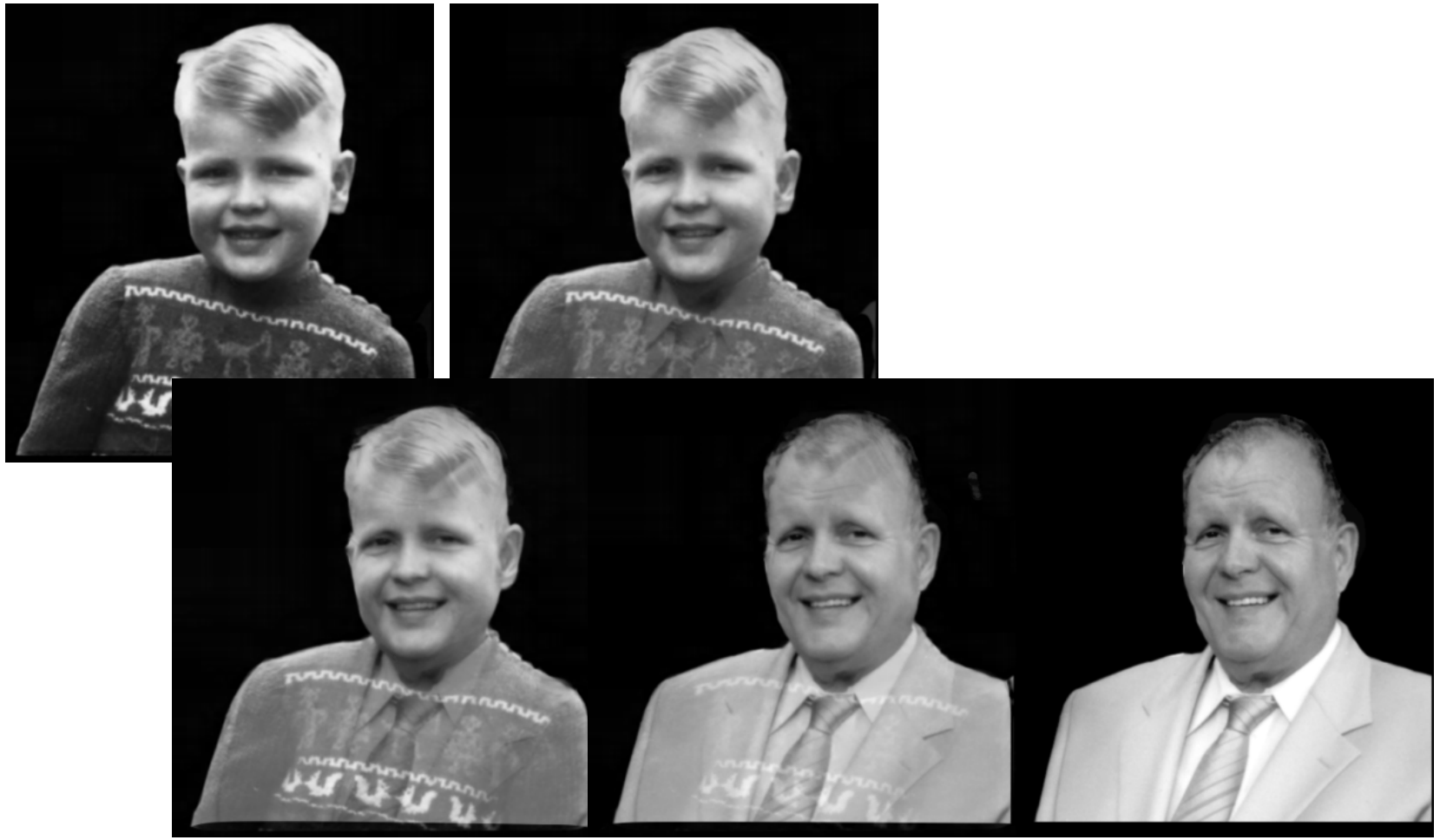
Example: Below the same person is shown as a child and as an adult. We want to fade smoothly between those two images.

2.5.4 Image morphing

Basic idea of image morphing

Simple interpolation of the gray values will result in an image somewhere in between the two but the result is not satisfying. On the next page we will see an example of what image morphing should look like.

2.5.4 Image morphing



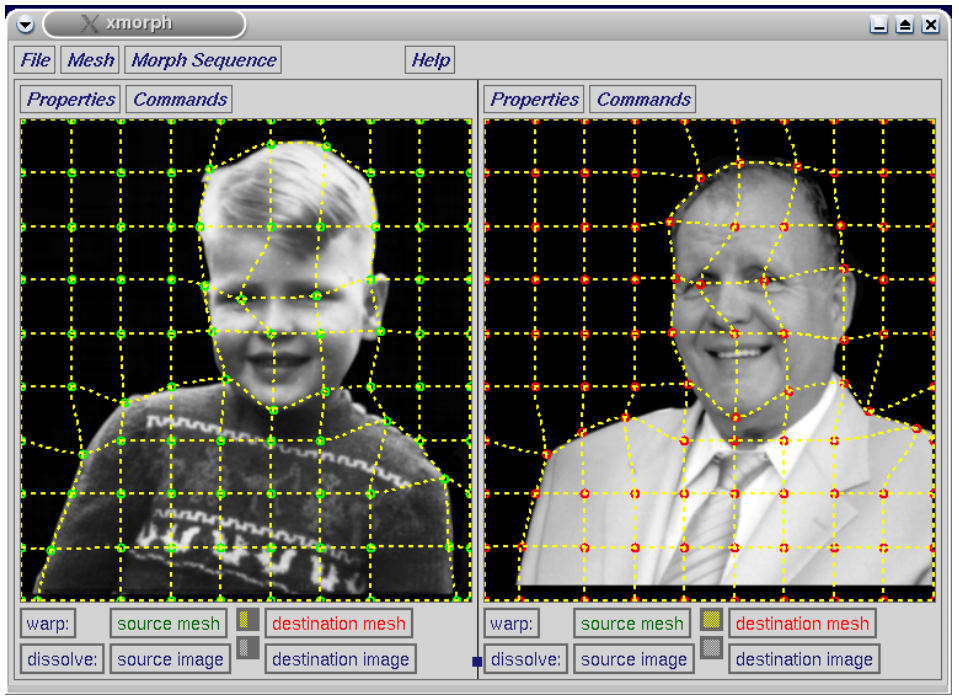
2.5.4 Image morphing

Using a regular mesh

Image morphing consists of interpolating the gray values while at the same time warping the image. This can be done using a regular grid. The user has to take care that characteristic features like the eyes are mapped onto one another.

Those approaches in which the user defines the correspondence between images are called landmark-based in the literature. They require quite a bit of user interaction but also lead to good results.

Image-based approaches, on the other hand, try to find corresponding points and lines in both images themselves. They require less user interaction but do not always find good matches from one image to another.



2.5.4 Image morphing

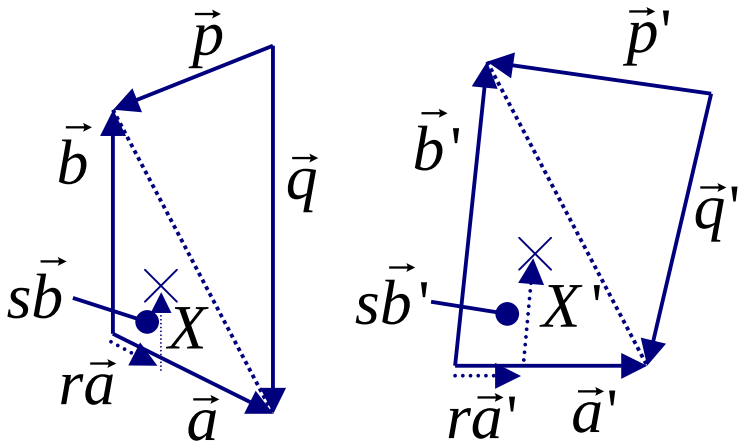
Using a regular mesh

Each pixel from one cell of a target image has to be mapped to a pixel on the corresponding source cell. First, we linearly combine a location X using two sides a and b.

The scalars r and s are defined in the interval [0,1]. They define the mapping of X‘ in the source cell.

X will be in the lower triangle if (r+s) < 1. Otherwise we have to use the vectors p and q for the linear combination of X in the upper triangle.

Note that it is the target image in which every pixel is considered exactly once. But many target pixels might be mapped to the same pixel in the source image if pixel locations are integer values.



target cell source cell

2.5.4 Image morphing

Linearly combine location X

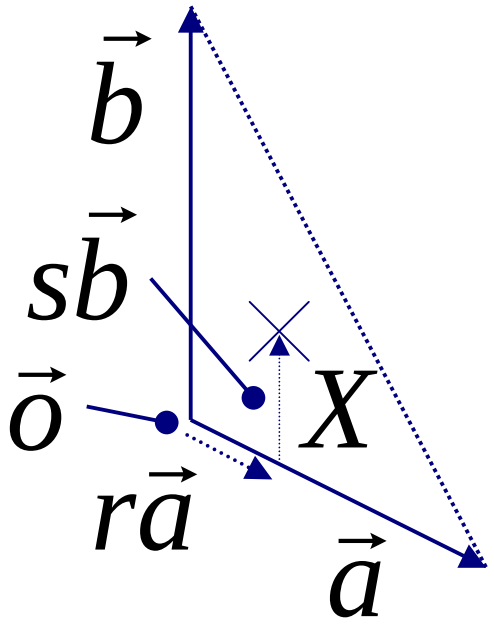
$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} o_1 \\ o_2 \end{pmatrix} + r\begin{pmatrix} a_1 \\ a_2 \end{pmatrix} + s\begin{pmatrix} b_1 \\ b_2 \end{pmatrix}$$

$$(\mathrm{I})\ r = \frac{x_1 - sb_1 - o_1}{a_1} \text{ for } a_1 \neq 0$$

$$(\mathrm{II})\ s = \frac{x_2 - ra_2 - o_2}{b_2} \text{ for } \mathrm{b}_2 \neq 0$$

$$(\mathrm{II}) \text{ in } (\mathrm{I}):$$

$$r = \frac{b_2(x_1 - o_1) + b_1(o_2 - x_2)}{b_2 a_1 - b_1 a_2}$$



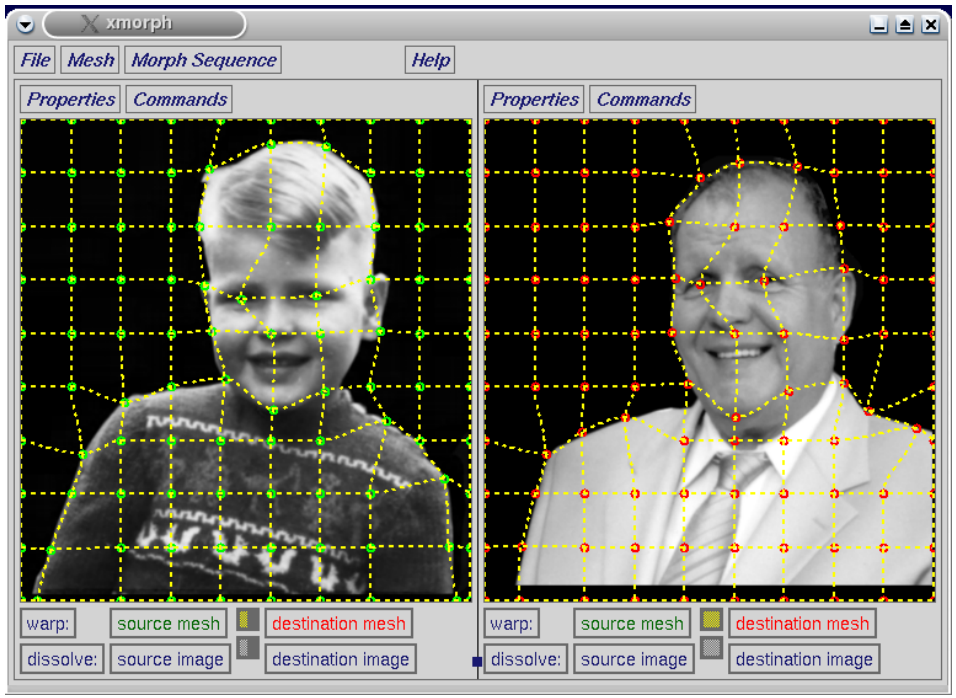
2.5.4 Image morphing

Using a regular mesh

The drawback to using regular meshes is that in many regions we have too many control points (like in the background). In other regions such as in the face region, there is a lack of degrees of freedom.

Want to practice morphing yourself?

Try xmorph or the more advanced XMRM v2.0 (both programs are free software).



2.5.4 Image morphing

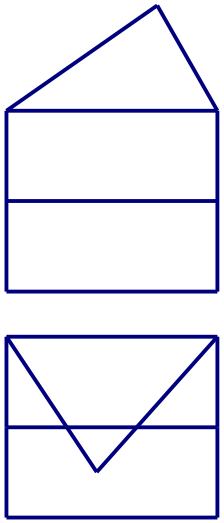
Using arbitrary meshes of triangles

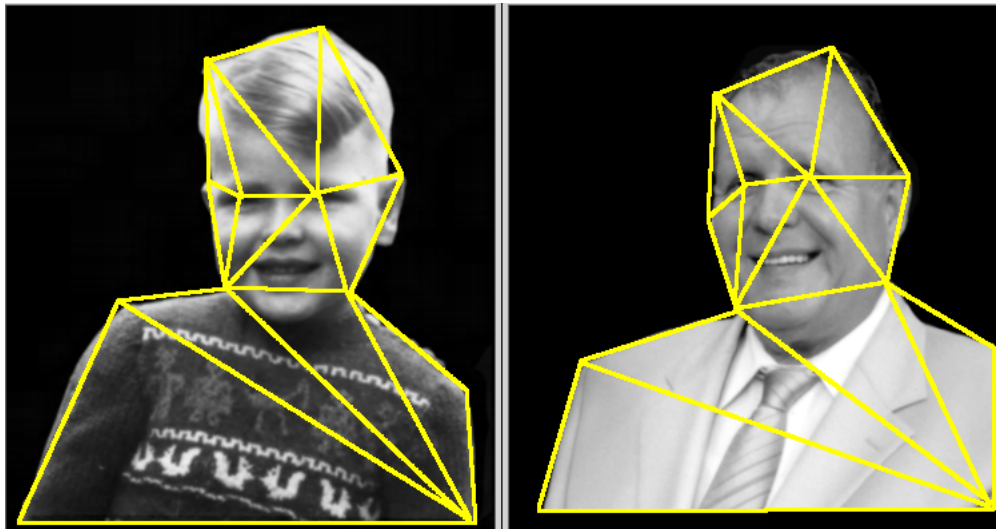
A solution to the grid problem is to allow arbitrary meshes of triangles. The user can now choose the granularity of the control points in different regions himself. Note that special care has to be taken that the user preserves the validity of the mesh. The example on the left side shows an invalid target mesh.

To achieve valid meshes the kind of linear interpolation from the previous page can be used.

source mesh

target mesh (invalid)





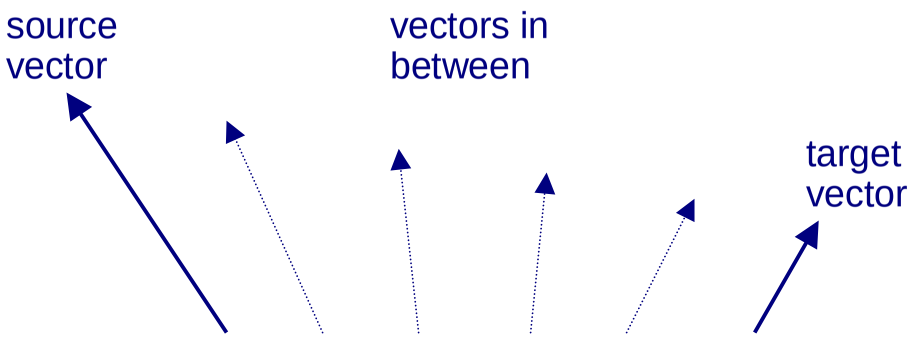
Example of a valid source and target mesh

2.5.4 Image morphing

Using arbitrary vectors

A more advanced approach gives the user even more degrees of freedom. He may choose to place single vectors all over the image without any constraints (Thaddeus Beier, *Feature-Based Image Metamorphosis*, Siggraph 1992)

An example is given on the following page.



2.5.4 Image morphing

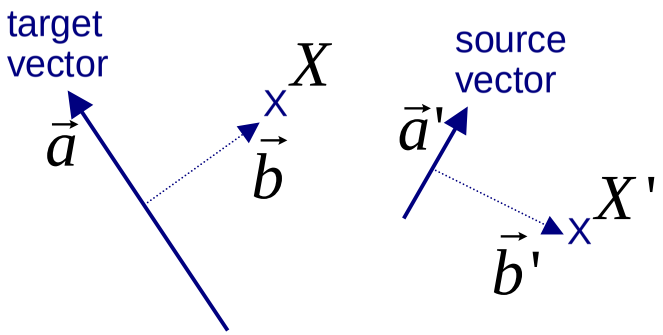
Using arbitrary vectors

2.5.4 Image morphing

Using arbitrary vectors

How does a pair of vectors in the source and target images influence the image?

Vectors a and a‘ are defined by the user. Orthogonal vectors b and b‘ are calculated respectively. a and b define a local coordinate system in the target image, while a‘ and b‘ define the corresponding CS in the source image. A point X is then expressed as a linear combination of a and b in the target image. The coefficients of the linear combination define X‘ in the source image.



$$|\vec{b}| = |\vec{b}'|$$

Most implementations keep the length of b equal to b‘ so that a scaling of the image is only allowed along vector a. The decision not to scale b‘ according to the ratio between a and a‘ is only based on empirical results.

What happens to the image if the user defines more than one vector?

2.5.4 Image morphing

Using arbitrary vectors

How do two pairs of vectors influence the image?

In the target image a point X can be expressed by two different coordinate systems. Usually this will result in two points X1 and X2 in the source image that are not equal.

$$X = \left(1 - \frac{s_1}{s_1 + s_2}\right) X_1 + \left(1 - \frac{s_2}{s_1 + s_2}\right) X_2$$

The final point X‘ is obtained by means of a weighting function. In the suggested function above X1 becomes dominant in the sum the more X converges against vector a in the target image.