Generating Storyboards based on Natural Language Descriptions

Generierung von Storyboards auf Grundlage von natürlichsprachlichen Beschreibungen
Bachelor-Thesis of Pavel Rojtberg
August 2009

Supervised by Prof. Michael Goesele, Prof. Iryna Gyrevych, Joachim Caspar

TECHNISCHE UNIVERSITÄT DARMSTADT

Computer Science Department
GRIS and UKP

Contents

List of Figures

Abstract

This work explores to what extent it is possible to automatically generate films from textual descriptions formulated in natural language today. For this purpose a prototype implementation is presented that explores possibilities and current limitations. The prototype generates a multimedia presentation using photographs from the web for visualisation and speech synthesis for sound.
Thereby the precision of flickr and google images as image providers is evaluated and improving the coherency of the results by user input and post processing is discussed. Finally this work gives suggests directions for future improvements by discussing the current bottlenecks.

1 Introduction

In the following chapter the automatic creation of storyboards will be motivated, then the goals of this work are described and some terminology used in this work is introduced. Finally there is an overview of the proposed method.

1.1 Motivation

In traditional film production the creation of storyboards is an intermediate step before the film is actually shot. They are used as a simple representations of the film to get an overview of the scenes and to spot possible problems beforehand. Storyboards are also the first stage in film production where the complete story is visualised in a meaningful way. Therefore it makes sense to choose them as the first step when trying to automatically visualise a story.
Typically storyboards are created directly from the script. They consist of a series of hand drawn sketches (see figure 1.1) which try to capture the gist of the scene in a few keyframes.

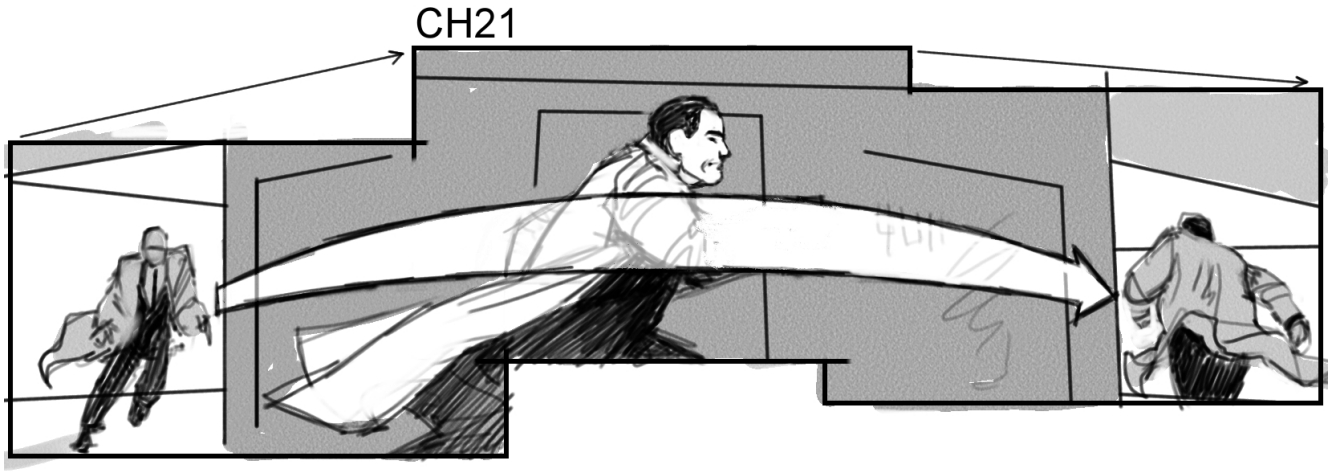

Figure 1.1: A typical storyboard image (From Goldman et al.[3])

A keyframe is an image of the scene which in some way represents the particular scene. It can show the subject of the scene in a representative pose or at an extrema of movement. Generally a keyframe visually summarises an aspect of the scene. In storyboards keyframes are usually annotated with arrows that visualise the movement of the subject and the movement of the camera. This is done by using bold three dimensional arrows for the subject and thin simple arrows for the camera. Storyboards are manually created, which is a time consuming process. Therefore they are drawn as sketches with only as much detail as necessary, which makes storyboards also an abstract representation.
In the following the traditional workflow of creating storyboards from textual representation will be called *forward storyboarding*. Because of the abstract view of the film and the summarising character of the storyboards, there are attempts to use them for representing films when processing them by computer. These methods usually try to generate a storyboard from an video sequence which is the contrary way compared to the historical origins of storyboards and therefore can be called *reverse storyboarding*.
In contrast, this work will try to automate the forward storyboarding approach which means generating them from a textual description. As generating a complete film is out of scope for this work, a photomatic

visualisation is used instead. A photomatic[13] is a series of photographs edited together and complemented with sound-effects and a voice-over. Photomatics can be seen as an extension of storyboards as they add more details and a timing aspect, but are still an intermediate step instead of the final product.

1.2 Goals

The goal of this work is to create a prototype which generates storyboards from a textual description given in natural language. Furthermore the storyboards shall be visualised with a film-like multimedia presentation.
This prototype shall be seen as the first step towards the ultimate goal to fully automating the creation of films from textual representation, like books. As just the first step there is no demand for practical results. Instead the prototype and this work as a whole will be explorative and will try to answer the question of how far we can get today. Furthermore it will explore current limitations and bottlenecks and give hints where future work would be most productive.
The prototype will use web image providers like flickr as a data source for generating the storyboards. As a contrast to existing backward storyboarding approaches the aim of this work will be data expansion and not data reduction, therefore the abstracting aspect of storyboards will not be considered.

1.3 Terminology

Scenes

Scenes as used in the context of this work refer to a continuous interval of time in the story, where locality does not change much. This may be either the case when the subjects do not move at all or move inside the same scenery. One can think of a driving sequence, where the subject of the scene is the car and the scenery is a town. As soon as the car leaves the town and the locality changes, the scene changes as well. Scenes are represented by a set of images in this work. This means that it is also possible to use none or more than one image to represent the scene. In any case the choice of representing scenes by images means temporal sampling of the scene. Therefore the selected images have to be representative.

Keyframe

A keyframe is an image which is representative for a given scene. This means it shows a key moment of the scene, which can be any of the following

- representative subject of the scene. A subject which is visible in the scene most of the time.
- representative set of subjects. Subjects which take part in the activity of the scene.
- representative pose. An extrema of motion which captures the activity described by the scene.

1.4 Overview

The process of automatically generating films from textual descriptions using an a priori created set of *scenes* can be seen as a pipeline consisting of three essential stages: content extraction, scene retrieval and scene adaptation (Figure 1.2). Because the prototype described in this work is restricted to using images as scenes it can be seen as just one instance of the general process.

In any case the input text has to be first processed in order to gain information like the scenery or the actors. This stage will be called content extraction and the method used in the prototype is explained in chapter 2. After there is knowledge about the content of the text, queries can be formulated to retrieve relevant scenes from the database of a priori created scenes. This stage will be called scene retrieval and

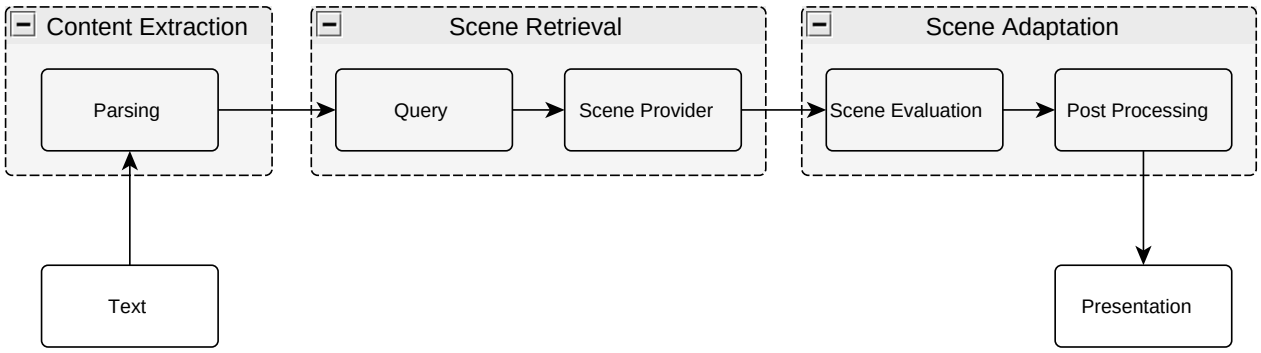

Figure 1.2: Pipeline for generating films from text

is described in chapter 3. The prototype implementation uses web image providers as the database of scenes. Because the database of available scenes is essential for good results, two image providers, flickr and google images are compared and evaluated.
Since the scenes are created before the content of the story is known, the likelihood of getting a perfect scene for the story is low. Therefore the retrieved scenes will have to be adapted in some way. This stage will be called scene adaptation and is discussed in chapter 4. As the database is assumed to be large yet not containing the perfect scene, it is also assumed that there will be several equally suitable scenes retrieved. Therefore the retrieved scenes will have to be evaluated in order to determine their content. With knowledge of the text content and of the scene content post-processing can then be applied in to perform the actual adaptation. After this stage there is enough information to present the results to the user as a film or as done in the prototype as photomatic, which is discussed in chapter 5.

While the pipeline described above contains the essential logical steps it does not fit well the pipeline actually used by the prototype discussed in this work (Figure 1.3).

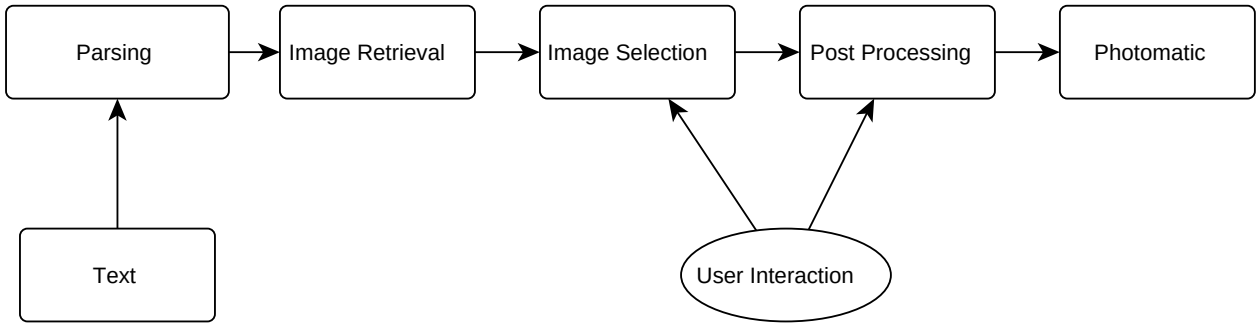

Figure 1.3: Concrete pipeline used in prototype

Compared to the logical pipeline the setup basically remains the same; only the abstract parts are instantiated for the use of images as the scene representation, which also leads to the choice of photomatic as the presentation format. However there is a change in the image evaluation step. In the prototype implementation the task of evaluating and selecting a coherent set of images is done by the user. Therefore the step in the concrete pipeline is called selection rather than evaluation. Because the evaluation is not done inside the system, there is also no way to automatically select an appropriate post-processing. Only the user has the necessary information and thus user interaction during the post-processing step becomes necessary as well.
A rather small change is the placement of the presentation step, which is called photomatic in the con-

crete pipeline, on the upper level. This is done to emphasise the greater importance dedicated to it in this work.

2 Content Extraction

Ideally the semantics or the meaning of the text should be extracted from the source text in this stage. However formulating semantics in a structured way is a big task on its own and the interfaces which will be used for querying in this work do not support the notion of semantics. The interfaces rely on keyword based queries instead.
Therefore it is enough to extract sensible keywords from the text in this stage, which can be achieved by examining the sentence structure. The sentence structure can be automatically extracted by constructing derivation trees(parse trees) for the given sentences (Figure 2.1).

```
(ROOT
  (S
    (NP (NNP Hansel)
      (CC and)
      (NNP Gretel))
    (VP (VBP are)
      (NP
        (NP (DT the) (NNS children))
        (PP (IN of)
          (NP (DT a) (JJ poor) (NN woodcutter)))))
    (. .)))
```

Figure 2.1: The parse tree of "Hansel and Gretel are the children of a poor woodcutter."

Derivation trees are constructed by application of a context free grammar(CFG) to a start symbol until only a sequence of terminals remains. The terminals represent the words of the sentence. Since there are many ways to create a parse tree for a given sentence, out of which only a few actually make sense, a probabilistic CFG(PCFG) is used to find the most likely parse tree. The following explanation and notation is based on the book *Speech and Language Proceessing*[10].
A PCFG consists of ordinary CFG production rules $(n \rightarrow \beta)$ supplemented by probabilities $P(n \rightarrow \beta)$ in the way that

$$\sum_{n \in N} P(n \rightarrow \beta) = 1$$

basically this means that n is exhaustively described by its outgoing production rules. Now a parse tree T for a sentence S can be rated by its total probability

$$P(T,S) = \prod_{n \in T} P(r(n))$$

The parse problem then boils down to computing the most likely parse tree $\hat{T}$

$$\hat{T} = argmax_{T \in \tau(S)} P(T|S)$$

In case of natural language parsing, the probabilities for the production rules can be estimated by analysing sentences annotated by parse trees. These parse trees encode information about the sentence structure and part of speech of the sentence.

Databases for sentences annotated with their parse trees are called treebanks. One popular example is the Penn Treebank[1], which contains English sentences from newspapers like the Wall Street Journal.
A lexicalized PCFG has extended production rules, which not only contain information about what will be produced using the production rule, but also which terminal(word) will be included in the produced sub-tree. This additional information is called lexical annotations of the grammar. It helps resolving ambiguities in sentence structure, where two different parses would have the same probability otherwise.

2.1 Related Work

Klein and Mannig[1] show that by Tag Spliting and head annotations one can bring the accuracy of an unlexicalized Parser to 86.36% (precision/recall over the Penn Treebank), which is comparable to state of the art lexicalized Parsers[12]. Furthermore they show[2] that by tag splitting the general performance can be improved. The idea is to factor the typical lexicalized PCFG into a lexicalized dependency part and a unlexicalised PCFG and then recombining them. This not only reduces the total system complexity, but also allows finding the best parse using the A^* shortest path algorithm.
This brings the complexity of parsing algorithms from $O(n^4)$ down to $O(n^3)$.
Furhtermore Klein and Mannig provide a reference implementation in form of the Stanford Parser, which uses the algorithms introduced by them.

2.2 Usage of parse trees

The Stanford parser is also used in this work to create parse trees. It is possible to use the stanford parser with lexicalized as well as non-lexicalized grammars for different languages. In this work it is used in conjunction with a non-lexical PCFG based on the Penn Treebank, since this gives the best performance. However this also requires the source texts to be written in English.
Each node of the parse tree represents syntactical information of the source sentence. Top-level level nodes encode how many sub-sentences, which are tagged with S are in the top level sentence, tagged with ROOT. Leafs on the other hand encode the part of speech of the terminals (Table 2.1). The tag

Tag	Description	Example
CC	coordinating conjunction	and
DT	determiner	the
IN	preposition or subordinate conjunction	in
JJ	adjective	red
NN	Noun, singular or mass	child
NNS	Noun, plural	children
NNP	proper Noun	Gretel
ROOT	tree root	-
S	(sub) sentence	-
VBP	Verb, non-3rd person singular present	are

Table 2.1: excerpt of treebank tags

names are also derived from the penn treebank, since they are determined by the used grammar.
In this work the part of speech information is used to find information carrying words in the text. Additionally the partitioning information of sub-sentences is used, since the scenery can change on sub-sentence level. For example the sentence "The girl walks through the woods to deliver food to her sick

[1] http://www.cis.upenn.edu/~treebank/

grandmother." has a change of scenery on the sub-sentence level. In the first part: "The girl walks through the woods" the subject is a girl and the scenery are the woods while in the second part: "deliver food to her sick grandmother" the subject is the grandmother which is presumably not in the woods.
Due to the stable scenery inside a sub-sentence it makes sense to associate scenes with sub-sentences by only using the information from the associated sub-sentence.

As only the parse tree for the current sub-sentence is examined when collecting information for the according scene, there must be a certain amount of information carrying words in them to be able to capture the gist of the scene. The amount of information carrying words in relation the sub-sentence length will be called *information density*. A sentence with a low information density is for instance "then they went there" where information about the subject and the destination has to be deducted from the context. A sentence with a good information density is for instance "then the girl walked to the house", where "girl" and "house" adequately describe the scene.

2.3 Text Restrictions

Besides the above described information-density the association of only one parse-tree to one scene imposes also a second restriction which will be called *locality*. Locality as used in this work refers to the requirement of a scene to show only one scenery. Therefore all information carrying words in one sub-sentence have to describe the same locality. In practice this means that the certain text types work better than others:

Summaries

Summaries naturally have a high information density, since they are meant to contain much information while being short. Furthermore they are mostly chronological continuous which also fulfils the requirement of locality, which makes them a good source.

Dialogues and free speech

Although dialogues and free speech mostly have a high enough information density, they are usually not chronological. This makes them refer to remote objects and locations. While visualisation in this case is still possible the rapid change of scenery would break any relation to the content of the sentence.

Books

Since books consist of both chronological continuous summaries as well as dialogues and free speech they are generally not a good source. On the other hand dialogues mostly happen in a constant scenery. In the context of the book it would be possible to just show the scenery to the dialogue instead of visualising it. But currently there are no optimisations for this case in the prototype.

3 Scene Retrieval

In this chapter the process of visualising scenes according to the information from the text is described. First related work is presented, then Google Images and flickr are discussed as image providers and their precision is measured. Finally the usage of tags as image descriptor is presented.

3.1 Related Work

Keyframes

Goldman et al.[3] present a method for semi-automatically creating storyoards by generating keyframes from a film which is already segmented into shots. Shots as defined in their paper are very similar to the definition of scenes made here. Their method requires the user to select a set of representative keyframes from the shot and do basic feature annotation. The feature annotation is done by marking still and moving parts of a shot. Out of that information full featured storyboards are generated. These are not limited to showing a keyframe, but also include annotation like motion arrows. Furthermore extended frames can be generated. An extended frame consists of several keyframes arranged together to show the pan of the camera.

While their method requires manual selection of keyframes to find extrema in motion, the method proposed by Assa et al.[4] can find these automatically. This is done by converting the motion into a low dimensional space and then searching for extrema in the resulting low dimensional curve. They also present a method for visualising several extreme points of a movement in a single image without overlapping and self occlusion by an expanded layout. While this method works on 3D skeletal models as well as on 2D images, it still needs manual foreground/background annotation when working on images.

Teodsio and Bender[5] follow a media transcoding approach from motion pictures to still images, where either high resolution images or stitches are created from a film sequence. In case of foreground motion like a moving person, representative poses have to be selected manually. If a timespan for the movement is selected the motion is represented by ghosts copies of the moving object.
But the generated summaries do not have to be stills. Pritch et al.[6] follow an approach where the time dimension is just scaled instead of being completely discarded. Therefore the result of their method is a film again, but compressed in respect to the temporal axis. This is achieved by automatically segmenting the video into events (foreground movements) and in case of spatial distinctness playing them back at the same time. The resulting temporal interleaving destroys the event order and is therefore only of limited use for storyboards, which require a correct timeline.

The above methods have in common that they follow the reverse storyboarding approach, i.e. generate storyboards or storyboard like summaries from already existing film sources. This work on the other hand follows the forward stoyboarding approach. These two approaches have only in common that they result in storyboards, therefore only the problem of summarising scenes for storyboards is actually related. The work which relates to automatic forward storyboarding most is by Blankinship et al.[11]. Their method allows comparison between a textual description and the final film based on it. It is based on the transcript of films embedded as closed captions in the signal of TV broadcast streams. The transcript is used to segment the film into shots which can be then compared to the original text. But the general aim of their work is to allow film editing by reordering the different shots.

Image Features

After selecting several suitable key frames it is necessary to select the most appropriate one automatically. To do so a group of relevant image features has to be identified, which then can be combined to an image descriptor. One popular descriptor is the gist descriptor[8], which tries to capture the information of an image a human would perceive in a very short timeframe, which is about 200ms. Since details areinfer generally not perceived in this timeframe it is describing the whole image as opposed to describing specific details of the image. Thus it belongs to the group of scene descriptors. For instance it contains the organisation of colour blobs. This is done by producing a low resolution version of the colour channels of the image. Since the gist descriptor is low dimensional, it can be used for quickly finding a set of similar images.
An application of the gist descriptor is described by J. Hays and A. Efros[7]. They use it to pre-select a set of similar images from a large database of images. These images are then processed by more computational intensive algorithms. The general method described in their paper uses a set of similar images to fill gaps in a different image. It is achieved by finding semantically equivalent images and combining them in a non obtrusive way.
Although the gist descriptor finds semantically equivalent images, it is only of limited use for storyboard generation. The problem is that the definition of two semantically similar images is different in this case. While in [7] it means that two images look the same while showing different objects, in this work two semantically similar images may look different as long as they are showing the same object. Therefore an object descriptor is better suited here.

3.2 Image Providers

The way queries have to be formulated first of all depends on how queries can be formulated i.e. the available querying API. Therefore we will look in the following at several possible image providers. A good image provider is expected to have the following properties:

- allow using as much as possible of the information we gained in the previous step for the query. This allows a fine grained selection of the images.
- high image resolution, so images are suitable for full screen displaying
- allow retrieving a large set of images, so data mining is possible

Generally realistic photographs are preferred over sketches, since the latter are often restricted to only one object and generally do not provide the impression of a complete scene. For evaluation flickr and google images as image sources are discussed. In this context the use of tags as image descriptors is also discussed and whether or not tag search is more suitable than traditional full text search.

Google Images

Actually Google Images is not an image provider - it rather offers a search interface for images located anywhere on the web. Therefore it is interesting since it has access to a very large set of images. Google images offers a public API[1] which provides free-text search and allows restricting the results by size, type or colour. (Table 3.1) Especially the latter option is interesting since it provides automatically extracted information from the image content. On the other hand the API has no option for ordering the results, so one can not prioritise one of the restriction parameters.

Without specifying the image type there is a very high scatter between the results as some of them are drawings, some are symbolic photographs and some are real world photographs. But even when restricting the image type to photographs using the API the scatter is still high.

[1] http://code.google.com/apis/ajaxsearch/documentation/reference.html#_fonje_image

Option	Description
Image Size	as in resolution
Image Color	greyscale/ colour or dominant colour
Image Type	Face, Photo, Clipart, Drawing
Text	Free text search

Table 3.1: Query options provided by Google Images

Figure 3.1 shows a query with type restricted to photos and image size to large, where the results still contain many sketches. The images are available in their original resolution, but since the images are stored on third-party servers, it is possible that the cache used for searching is out of sync and the images found are not available any more.
Furthermore the number of results using the official API is limited to 64 in total, which is far below the size required to perform data mining on them.

flickr

Flickr is a web photo community which provides image hosting to its users and therefore is an image provider. This also ensures that unavailable images are not listed in the search index. Furthermore flickr allows many ways to interact with the images like commenting them, tagging them or marking points of interest. It also provides a public API[2] which offers many parameters to influence search results. It is possible to search in tag mode or in free text mode. Searching in free text mode includes related texts like comments on the image, the image description, the image title and tags associated with the image. Tag based search on the other hand includes only the associated tags for searching. Furthermore the API allows to specify whether the set keywords should be treated as a conjunction or a disjunction. It is also possible to sort the results by relevance, interestingness (number of hits) and date, which allows prioritising one of the restriction terms (See Table 3.2).

Option	Description
Image Size	as in resolution
Image Type	Photo, Screenshot, Other
Date	date taken, date uploaded
Sort by	relevance, interestingness, date
Text	Free text search
Tags	Tag search
Location	GPS coordinates

Table 3.2: Excerpt of query options provided by flickr

The sorting option *relevance* is only relevant when searching for a disjunction of keywords. In this case it sorts the results by the number of found keywords.
Looking at the API flickr is more flexible and therefore allows more fine grained control on the results while Google is easy to use but hides many interesting options. Although Google automatically infers some information from the image like the most prominent colour and whether a face is visible, the API does not provide much to control this parameters. Flickr on the other hand does not infer any informations automatically but instead relies on manually added tags, which, while being less reliable, provides a very fine grained control of the results.

[2] `http://www.flickr.com/services/api/flickr.photos.search.html`

Figure 3.1: Google Images results for the Query "animals woods trail breadcrumbs"

Therefore the possibility to search only by tags (Figure 3.2) as an alternative of doing a full text search (Figure 3.3) is a good option.

3.3 Precision of Results

To compare the image providers and different search options, the precision of the retrieved results is measured.

$$\text{precision} = \frac{|\{\text{relevant images}\} \cap \{\text{retrieved images}\}|}{|\{\text{retrieved images}\}|}$$

To use realistic queries at least two keywords are used per query. This assumption can be made since the example texts "Little Red Riding Hood" (Wikipedia Summary) and "Hansel and Gretel" (Wikipedia Summary) contain 2.2 nouns per sub-sentence on average. On account of this an image is considered relevant when it shows at least two of the keywords from the query and is a photograph. This approach on relevance also considers that images with many objects from the source text have a higher semantic connection to the source than images showing only one object - even if the object is the subject of the scene. Since the evaluation is done manually, there is still a fuzziness factor due to subjective judgement. Often the recall is also stated together with precision:

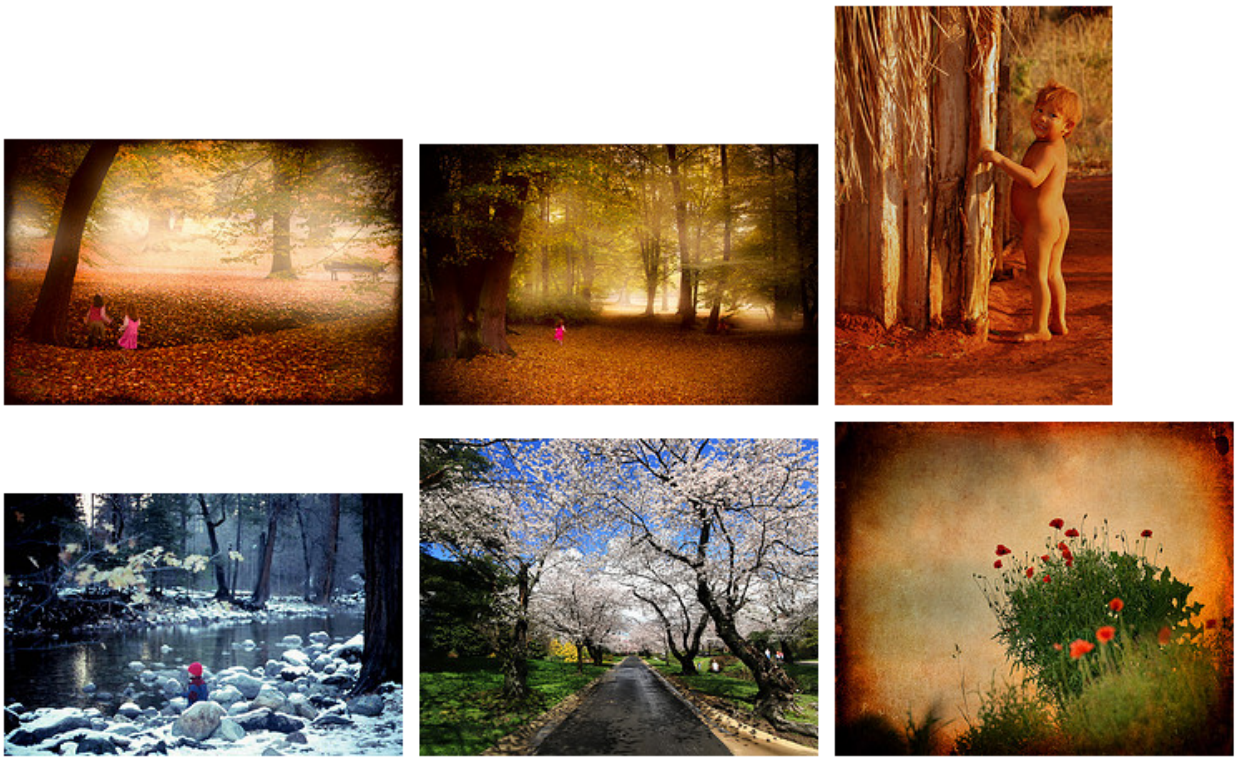

Figure 3.2: Flickr results for the query "animals woods trail breadcrumbs" using tag search

$$\text{recall} = \frac{|\{\text{relevant images}\} \cap \{\text{retrieved images}\}|}{|\{\text{relevant images}\}|}$$

but in this case it is not possible to compute it accurately since it requires an a priori knowledge of the set of relevant images, which is not given because this set is almost infinitely large in our case. Although one could approximate this set by computing the union of all relevant results retrieved by all image providers, it would require sorting out of duplicate images manually and still would have a strong coupling to precision. For this reason recall is not considered.
Figure 3.3 shows some results from the complete evaluation which was performed with the queries

- animals woods trail breadcrumbs
- house gingerbread candies sugarwindows
- children forest
- starvation woodcutter wife
- wolf body stones

and the search options *disjunction* and *order by relevance* for flickr and *large image size* for google. Both providers were configured to return a set of 30 photographs.

Additionally the graph in figure 3.4 shows the development of precision with an increasing size of the result set. This is of interest as it shows how fast a matching image can be found in the result set.

As we can see flickr tag search has the best total precision and also the best precision curve. In 80% of the cases already the first image is relevant according to the criteria mentioned above, but the precision goes down quite fast to 40% with increasing result size due to increasing scatter. Google Images just

Figure 3.3: Flickr results for the query "animals woods trail breadcrumbs" using full text search

Image Provider	Query	Precision
Google Images	animals woods trail breadcrumbs	7%
	children forest	3%
	house gingerbread candies sugarwindows	37%
flickr full text search	animals woods trail breadcrumbs	37%
	children forest	17%
	house gingerbread candies sugarwindows	33%
flickr tag search	animals woods trail breadcrumbs	70%
	children forest	40%
	house gingerbread candies sugarwindows	50%

Table 3.3: Precision of the image providers

starts at 40% precision and goes down to 20%, while flickr text search stays more or less constantly at the precision of 20%. Overall flickr tag search has the best precision as well as the best precision development with increasing result size therefore it is chosen as the image provider in this work. Out of Google Images and flickr text search, the former is preferable due to the higher initial precision - although both converge against the same precision with a result size of 30 images.

3.4 Tags as Image Descriptors

Looking at the above results it is obvious that the tag based approach is favourable for searching, although one might expect that including more related information in the search also leads to an increased likelihood to find the right image. Nevertheless in practice it actually decreases the precision, since it also finds just loosely related images and thereby increases noise. Lets consider the query "walk in forest";

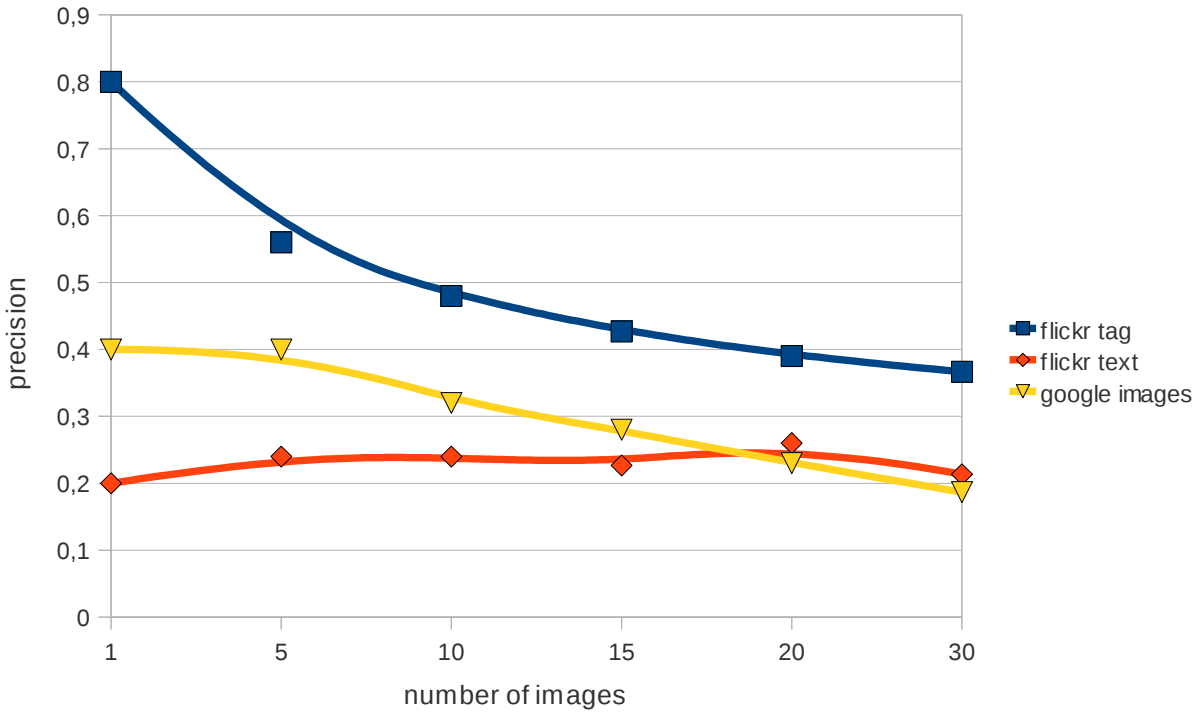

Figure 3.4: Average precision curve over the 5 tested queries

when using the full text search one might encounter an image of two people having dinner, where the author added the comment to the image "we had a great walk through the forest afterwards". Tags on the other hand directly refer to the image content and are added explicitly to describe it. Therefore the likelihood of a false positive is much lower. An image tagged with "walk" and "forest" is very likely to picture a forest and some people walking.
Since tags have to be added manually, there might be concerns that there are only few of them available per image. But a test with the query "girl woods" showed that the returned set of 30 images had 22.7 tags attached per image on average.

Figure 3.5 shows that tags adequately describe an image and even include information like "selfportrait" or "joy". This information one would usually expect to be only present in related sources, like comments.
One tag only describes one aspect of the image - in this regard tags can be compared to automatically computed object descriptors, where one image descriptor only describes one aspect of the image. For instance parametric object detectors can be used to find a bone inside the image - or the image can be tagged with "bone". Furthermore both computed descriptors and tags are not exhaustive - while an image showing a bone is not necessarily tagged with bone, a computed object descriptor might give a false negative for the same image. A difference between the two methods is the kind of information which is provided beyond the presence of an object: while object descriptors give the exact position of the object in the image, tags can describe properties which are non-trivial to compute like "happiness".
Although there is a reasonable amount of tags attached to images on flickr, they are not exhaustive and subjectively prioritised; for instance a picture showing a girl standing in front of a house will likely be tagged with "girl" and probably "house" but not "tree" or "street" even if those are pictured as well. The decision which tags are relevant for a image is made subjectively by the user. Therefore it makes sense to treat the query as a disjunction since otherwise many relevant images would be excluded. Instead one should sort the results by the number of matched tags. For this matter the option to sort by relevance on

- boy
- summer
- girl
- children
- woods
- joy
- happiness
- valentijn
- galefraney
- abigfave
- galefra
- alemdagqualityonlyclub

- selfportrait
- film
- girl
- hair
- polaroid
- woods
- branches
- brush
- 600
- tangled

Figure 3.5: Images with tags from the query "girl woods"

flickr is suitable.
Another observation is that an increase of the number of tags or other information attached to the image does not result in better search results but in greater noise. The main problem here is that all tags are equally weighted. Two images, one showing a close up of a tree and one showing a house and a tree in the corner, would be equally relevant judging just by the tag "tree". This basically ignores the importance of an object in the image.

Relevance of part of speech

There is no strict naming scheme for tags, but looking at the *all time most popular tags* on flickr (Figure 3.6) 92% of the tags are nouns, 8% are adjectives and none are verbs.

This is not surprising, since nouns are the part of speech which describe objects. Therefore they are essential for tag based queries. While using adjectives could lead to better global results, generally they decrease the precision of the results (Figure 3.4). This is probably caused by the subjectiveness in the decision whether an image is "sad" or not. Therefore they will not be included in the prototype implementation, but surely deserve a closer look. Verbs on the other hand have only little influence on the search result, but also decrease precision and are not being considered in the prototype either.

Figure 3.6: All time most popular tags on flickr

flickr tag search query	Precision
cake grandmother	57%
cake sick grandmother	47%
bring cake grandmother	53%
bring cake sick grandmother	47%

Table 3.4: Precision of flickr tag search with different queries

For the evaluation of the influence of different part of speech in tags a set of 30 images was retrieved for the given queries, the keywords were again treated as a disjunction and the result order was *relevance*. An image was considered relevant when it contained one of the nouns, since measuring the influence of the other parts of speech on the query was more important than measuring adequacy this time.

Used queries

Based on the above observations the queries are generated from nouns contained in the text which can be found by searching for nodes of the parse tree labeled as NN, NNS or NNP.
In order to meet the requirement of locality of a scene as described in chapter 2.3, the text is split into sub-sentences beforehand, so the queries are generated on the sub-sentence level. This also limits the number of nouns per query, which helps reducing image scatter by only specifying the main subjects. The query terms are treated as a disjunction due to the non-exhaustiveness of tags.

4 Scene Adaptation

After sending the query to the image provider we retrieve a set of suitable images. This happens on sub-sentence level, so we get an association between a sub-sentence and a set of images.
To visualise the story we now have to look at each sub-sentence and select an *adequate* subset of images. Therefore this chapter starts with the definition of adequacy which motivates the method of selecting the images. It is then discussed whether the selection method can be implemented automatically and how it can be done manually by the user.
Next it is discussed to what extent the results of the selection can be improved by post-processing. Finally sorting images locally is presented as a future improvement for the selection problem.

4.1 Related Work

E.H. Adelson and J.R. Bergen[16] introduce the plenoptic function. It is a continuous function describing the space-time defined as

$$P(\theta, \phi, t, \lambda, V_x, V_y, V_z)$$

which gives the intensity of the colour with the wavelength λ when standing at the position (V_x, V_y, V_z) and looking in the direction (θ, ϕ). It can be used to describe images rendering or panoramas as sampling along one of the axis of this function: an image is described by restricting the view direction to a view angle and choosing a fixed value for all other dimension except for λ. A panorama could be represented by choosing a fixed t so the scenery is static and by choosing a fix viewer position (V_x, V_y, V_z). Only λ, θ, ϕ remain variable.

4.2 Adequacy

While adequacy can be judged intuitively we would like to get a measurable mathematical definition. Therefore we will look closer at the results of the previous step: after associating a set of images to each sub sentence the story is represented by the tuple S of sets

$$S = (s_1, s_2, \ldots, s_N)$$

where N is the number of sub-sentences in the text. When selecting exactly one image out of each set $s_i, i \in [1..N]$, this can be seen as a temporal sampling of the story using N samples. The image at each of the temporal samples describes the world from a position (V_x, V_y, V_z) looking in the direction (θ, ϕ). Putting this together to a continuous story-function S results in

$$S(\theta, \phi, t, \lambda, V_x, V_y, V_z)$$

which basically is the plenoptic function, which is not surprising since we are talking about a set of photographs.
But now we have a means to express the story mathematically. To describe adequacy we can now look at the difference of two story-functions; one based on the photographs from the real world and one based on the world as described in the text. The difference of this two functions would the be the difference of the according images at the same temporal sampling point. But since there is no unique solution for the

world as described in the text, we can not use this approach to find a perfect solution. Yet we can use this as the definition of adequacy by comparing the information given in the text with the information given in the images. In this work this will be called *consistency with the story* and include the following aspects

- is the described scenery shown?
- are the described subjects shown?
- is the described activity shown?

Furthermore we will need to ensure that the images are also *consistent with each other*. This is necessary because the image come from arbitrary source and there is no guarantee that they fit together. For instance two images showing a grandmother might show different persons while the story refers to the same persons. To ensure this consistency the following aspects are included in this work

- is a character represented by the same person across the scenes?
- are colour and greyscale images mixed together?
- do the images have the same "mood" (dark, happy) and does the mood match the story?
- are sketches and photographs mixed? do the images have the same level of detail?

This can be seen as a requirement imposed by the steadiness of the story-function. Asking for steadiness of the story-function in the first place makes sense since, the story function is actually a family of functions. One can think of colour and black and white versions of the same story when it is not explicitly stated in the text. Without requiring steadiness arbitrary jumping across this two versions would be permitted, which is generally not desired.

4.3 Automatic Selection

Automatic selection would require automatically deciding whether an image is consistent with the story. With the current approach of only describing the story with nouns this would require an object descriptor for each possible noun just to answer the question whether it is shown on the image or not. While this might be achievable under some restrictions it is certainly out of scope for this work.
Another more practical method would be always selecting the first image returned from the query since the likelihood of getting a good image is quite high when using flickr tag search (Figure 3.4). But then again there is no guarantee that the selected images are consistent to each other. Ensuring this consistency would require image and object descriptors. While one might choose the gist descriptor to describe the images one would again need an object descriptor for each object one possibly might encounter in the text.

Furthermore the measurement in chapter 3 was done without taking the consistency between the images into account. If we therefore assume the first image is not relevant because of missing consistency between the images while assuming that the other images might be consistent the precision graph (Figure 4.1) does not support the selection of the first image any more.

4.4 Manual Selection

As no suitable method for automatic selection is available for this work, the adequacy problem is solved manually. For this purpose the sets of images retrieved from the image providers are presented to the

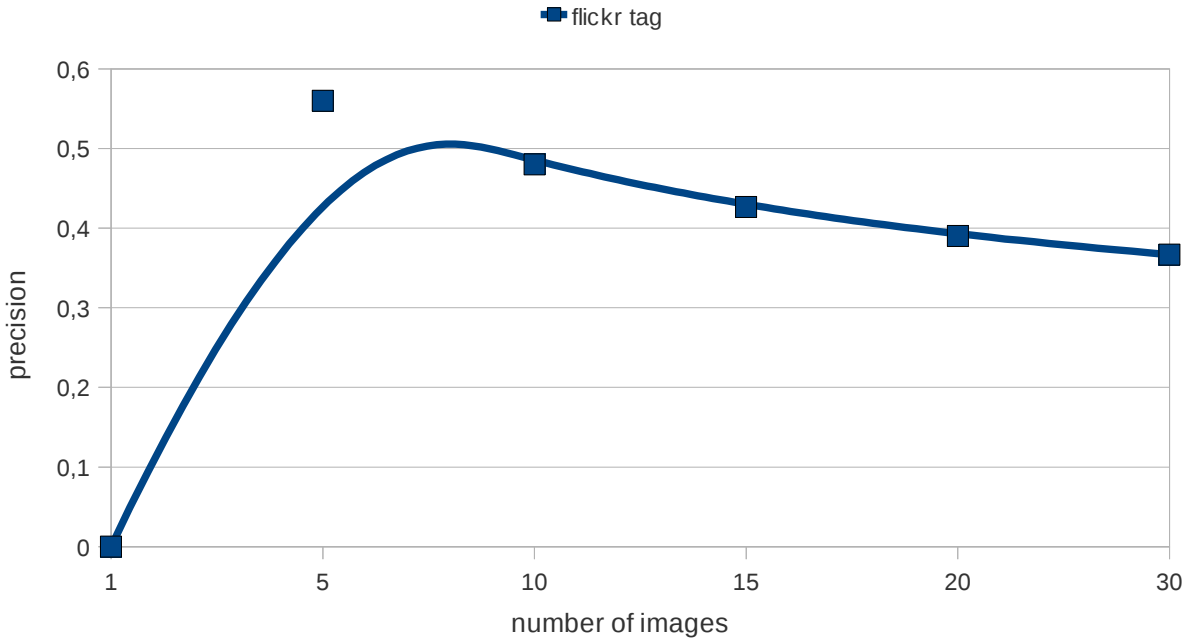

Figure 4.1: Average graph for precision assuming miss on first image

user, which then can select the most adequate subsets.There is no limit of one image per-sub sentance here, to allow the user to choose a more appropriate sampling. The term subset is used in the mathematical sense here.
When selecting the images the user has to ensure that the image is consistent with the story and that it is consistent with the other selected images.

Consistency with the story

To adress the first problem the user would have to review all images and consider all possible combinations, which is probably a bigger task than drawing the storyboard by hand. However the graph in figure 4.1 implies that only considering the first 10 results should already cover the majority of the relevant images. When also taking into account the theorem by Miller[18] that a person can only keep between 5 and 9 chunks in the short time memory only the first four images of the retrieved set are displayed (Figure 4.2).

This assumes that keeping the context of the sentence in memory takes up one chunk of memory. In case there was no good match among the first 4 images it is possible to subsequently request the next set of 4 images. Of course the question of consistency is subjectively decided by the user, but this is not necessarily a disadvantage as consistency is perceived subjectively anyway.

Consistency between the images

When it comes to consistency between the images, the user has only to ensure manually that the same characters and locations are used across the story. The other aspects mentioned above can be achieved by applying post-processing which is discussed in the next section.
Ensuring the usage of the same locations and persons across the story requires an overview over all selected images. Since this can easily exceed the number of the 7 recallable chunks, the prototype allows to display a preview of the images selected up to the current to get an fast overview again.
Futhermore it is possible to re-use an image already selected for a sentence for visualising a subsequent sentence (Figure 4.3).

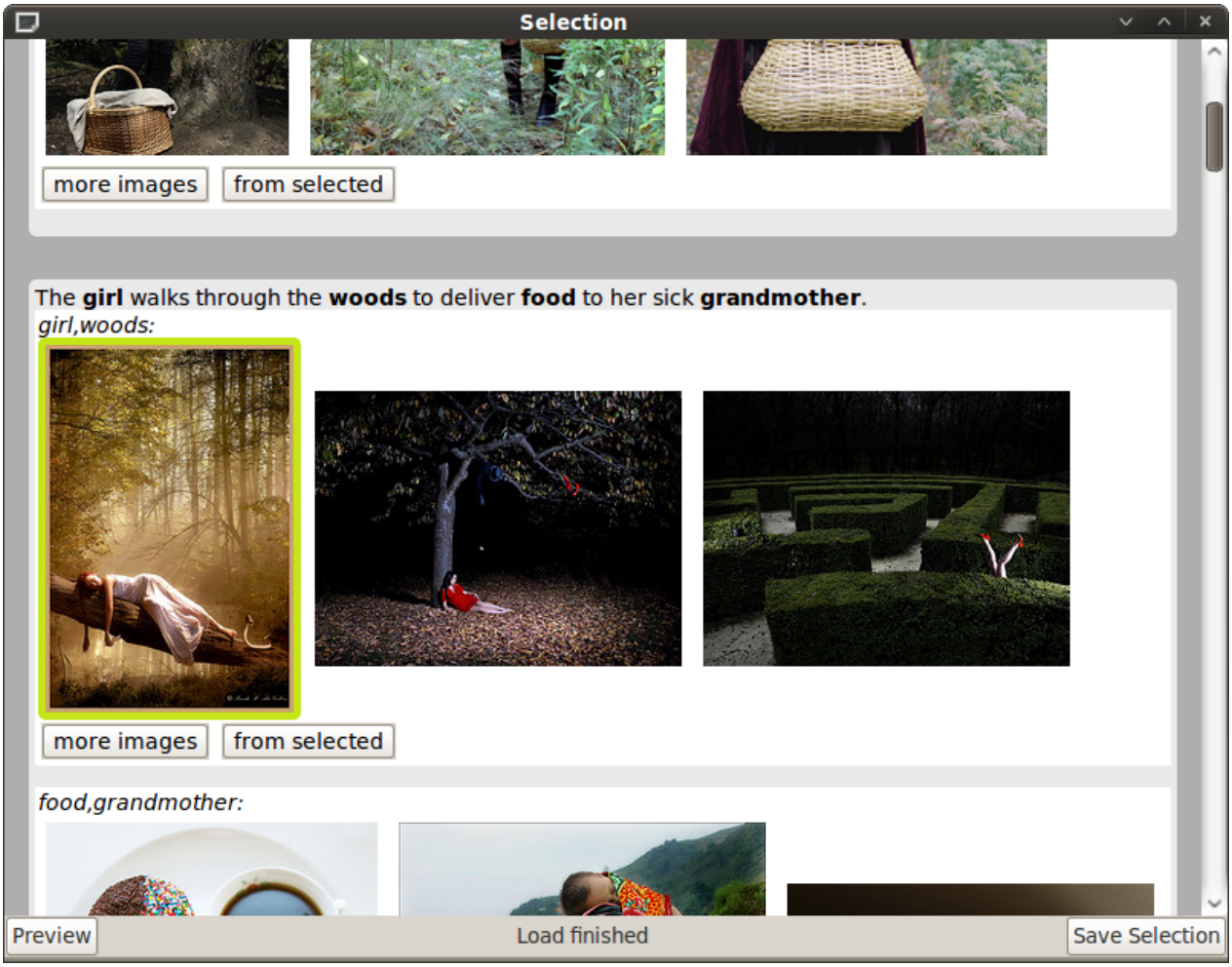

Figure 4.2: Interface for choosing relevant images

This helps the user to choose the same person for different sentences even if the extracted keywords differ. For instance in the sentences: "The story is about Little Red Riding hood (...)", "She goes through the forest (...)". When refering back to "Little Red Riding Hood" for the keyword "she", it is possible to manually create links between the sentences.

4.5 Post processing

While the consistency between images has to be ensured manually regarding the content of the images, it can be improved automatically by post-processing when it comes to the look of the images. The prototype implements post-processing effects to improve the consistency of the images regarding the aspects

- are colour and greyscale images mixed together?
- do the images have the same "mood" (dark, happy) and does the mood match the story?
- are sketches and photographs mixed? do the images have the same level of detail?

as listed in section in section 4.2. See table 4.1 for a overview of the effects and their possible effect on consistency.

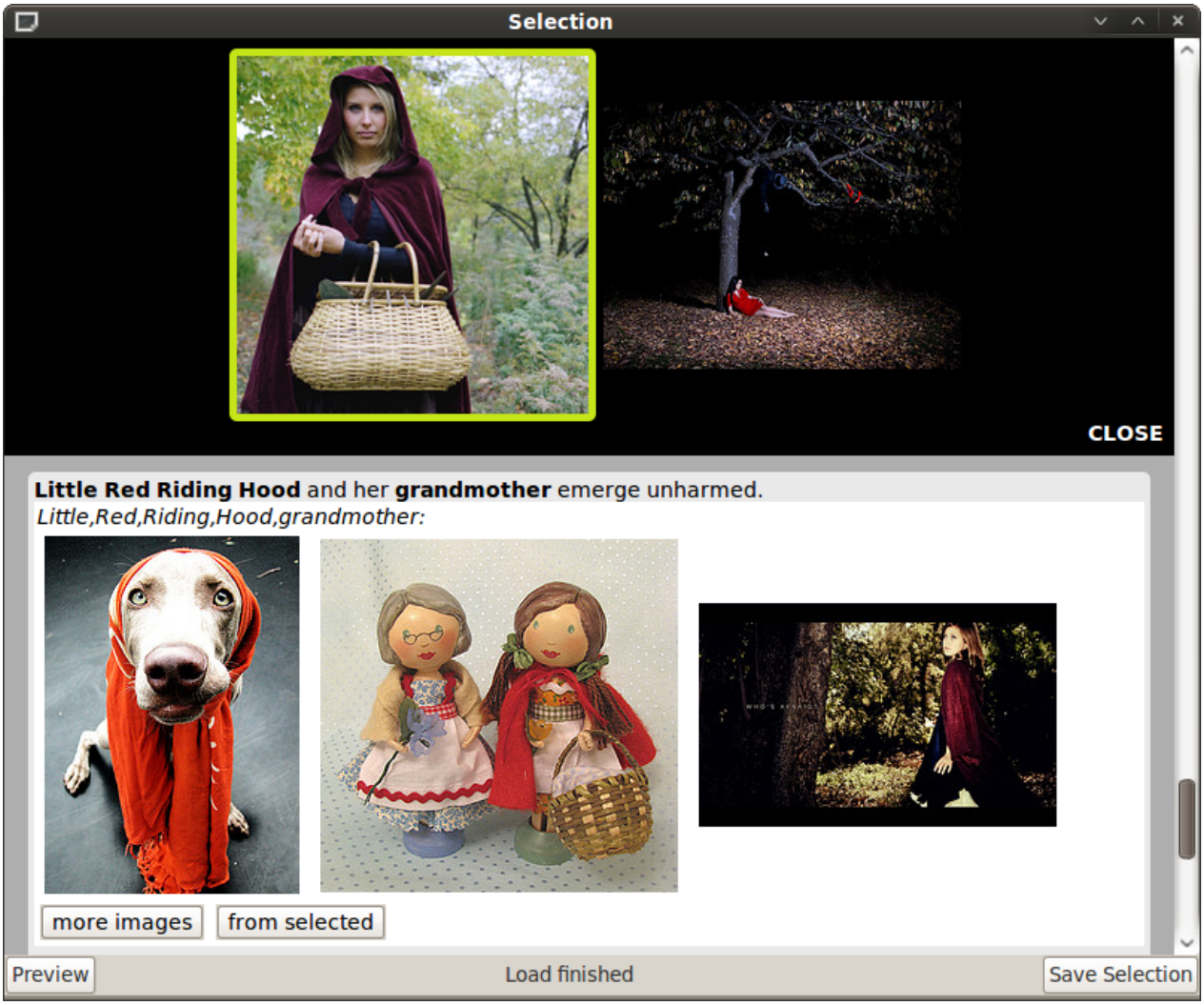

Figure 4.3: Interface for referring back to selected images

One simple post-processing effect which would improve the consistency in regard to the *colour descriptor* would be desaturating. By applying this point-operation to a set of colour and greyscale images the colour difference between them would be eliminated. Another effect of desaturating an image is that it is perceived to be more sad, while applying the bloom filter warms up the images, which is then perceived as more happy (Figure 4.4).

But there is no general rule how a post-processing effect is perceived - it depends on the context and the general style of the story. As can be seen in figure 4.6 the Bloom effect can also improve the consistency between images in comparison to the original (Figure 4.5).

While applying the sepia point-operation to this example (Figure 4.7) results a darker "mood" of the images. Although sepia is basically just a point operation which consists of desaturating and colourising the image.

Another interesting effect is non-photorealistic rendering, which controls the level of detail of the image. This can not only be used to ensure the same level of detail across the set of selected images but can also reduce the difference between two different persons used to represent the same character. This can be achieved by reducing the details to a level where the facial features are not visible any more and the character is thus perceived only symbolically.

The non-photorealistic filter is implemented as an edge preserving blur filter similar to the image space technique described by B. and A. Gooch [17]. The blurring component of the filter is responsible for

Effect	Description
Sepia	abstracts from different colours. matches fairytale style
Bloom	influences the mood of the image. Can be used for to create a *warm* feeling
Black & White	abstracts from different colours.
Non Photorealistic	changes level of detail

Table 4.1: Post Processing Effects

Figure 4.4: Mood influencing effects: original, Black & White, Bloom

choosing the scale of the image, but the scale which has to be chosen here depends on the content of the images which would have to be analysed beforehand and is not possible in the timeframe of this work. Therefore the prototype uses the simple symmetric nearest neighbour filter for this task, which gives good results for medium to large scales.

4.6 Local Sorting

While the above results show that it is possible to change how the images are perceived after their retrieval, the consistency improvement is achieved by information destruction. Another approach which would not destroy information is to retrieve a large set of relevant images and then to search inside it for the desired criteria. When using a large enough set of images this can also completely replace the image provider step which would allow more domain specific queries. These could then rely on complex object descriptors which are not feasible to compute for a web image provider due to their specific use case.
In order to evaluate this local search a combined approach is used, which first uses flickr tag search (see 3.4) to find the requested object and then a local search to find the requested colour.

Figure 4.5: Improving consistency: initial images

Figure 4.6: Improving consistency: Bloom Effect

The test is done by downloading a set of 1001 images tagged with "forest" and then sorted according to the similarity to the reference colour. Thereby the images are sorted twice; first by relevance and then by colour which ensures that the topmost image has both properties: the desired object and the desired colour.
The results (figure 4.8) show that this method allows to influence the mood of the image without data reduction; while sorting by black results in a dark scene, sorting by red creates an aggressive atmosphere.

The sorting by the similarity to a reference colour $\vec{c}$ is performed by integrating over colour difference at image point in the image area Ω. To normalise between different image sizes the resulting difference is divided by the size of the image area. Since the colour space is similar to the euclidian space $\|\cdot\|_2$ is used to compute the distance of two colours. The rating r according to which the images are sorted is then:

$$r = \frac{\int_\Omega \| I(p) - c \|_2 \mathrm{d}p}{|\Omega|}$$

As the images are stored discretely the integral reduces to a sum and the computational intensive square root in $\|\cdot\|_2$ can be left out since it is a monotonically increasing function so the computation reduces to:

$$r = \frac{\sum_{p\in\Omega} \langle p - c, p - c \rangle}{W \cdot H}$$

Figure 4.7: Improving consistency: Sepia Effect

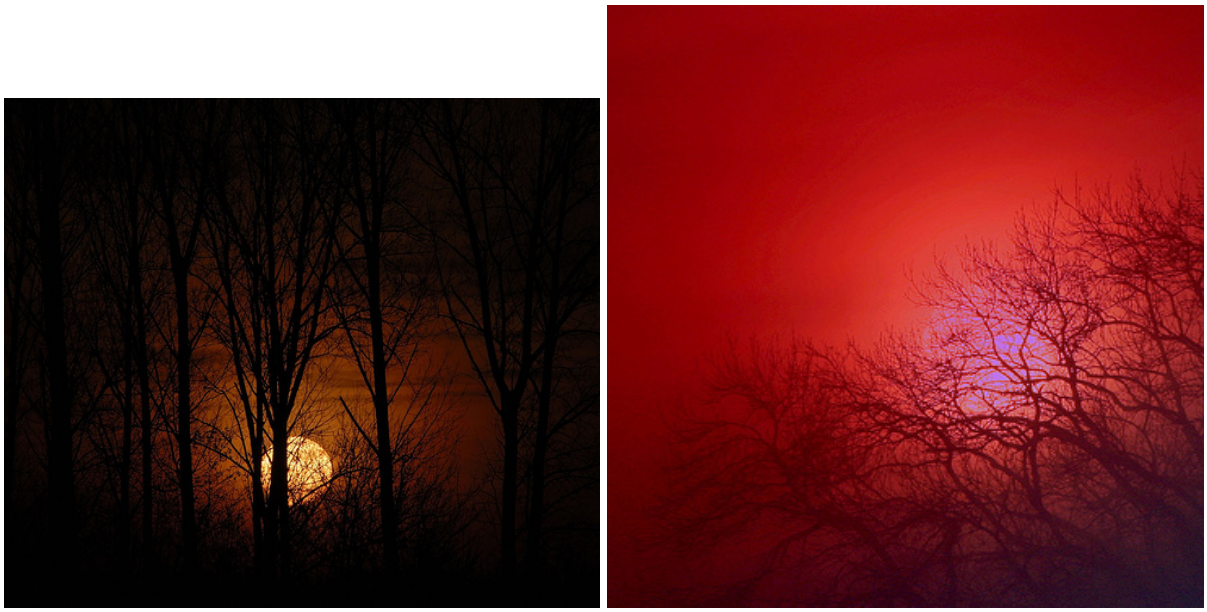

Figure 4.8: First results for local sorts by similarity to black (left) and red (right)

where W and H are the dimensions of the image. Using the simplified formula the sorting of the 1001 images can be performed in 10.9s[1].

[1] done on an Intel Core2 Duo, 1.83Ghz

5 Presentation

In this chapter the method used for presenting the results to the end user is discussed. This includes the visualisation of the images as well as composing the text.

5.1 Related Work

A. Girgensohn describes a algorithm[9] for laying out video summaries. This algorithm consumes a set of images representing the video and arranges them in a manga layout. The algorithm preserves the original image order and has an short runtime which allows real-time applications. Furthermore it allows weightening the individual images so important images are shown bigger.

In the area of speech synthesis statistical parametric systems based on Hidden Markov Models(HMM) became quite popular over the recent period according to Zen et. al[14]. Therefore they propose a HMM based model to generate speech. The model allows easy modification by transforming the HMM parameters and can be used with the Festival speech synthesis runtime.

5.2 Speech Synthesis

The festival runtime is also used in this work to synthesise speech. It allows the usage of different synthesis models whereas the diphone model is used as default. But this has non satisfactory results, therefore the HMM based model mentioned above is used. There are several voices available for using this HMM.
The prototype includes the following voices:

- a bright female voice
- a deep male voice
- a british male voice

this choice does not allow a fine grained control of the theme of a story by selecting a specific voice, but it allows to observe the influence a voice has on how a story is percieved.

5.3 Visualisation

The visualisation of the images is done by generating a simple photomatic. The necessary voice-over is created by synthesising the source text. This is appropriate in most cases since the prototype works best with summaries as discussed in chapter 2.
If no voice is selected sub-titles are displayed instead (Figure 5.1).

These are not visible otherwise to keep the user attention on the displayed images. To determine the timing of the images the length of the sentence is used after synthesized to speech. Furthermore it is padded by a delay of 200ms to control the speed the story is told. Without any pause between the sentences the story is told too hasty.

When more than one image is associated to a scene, which for instance is the case when there are two sub-sentences, the timeframe (Figure 5.2) is equally distributed between the two images. In the

Figure 5.1: Presentation application showing subtitles

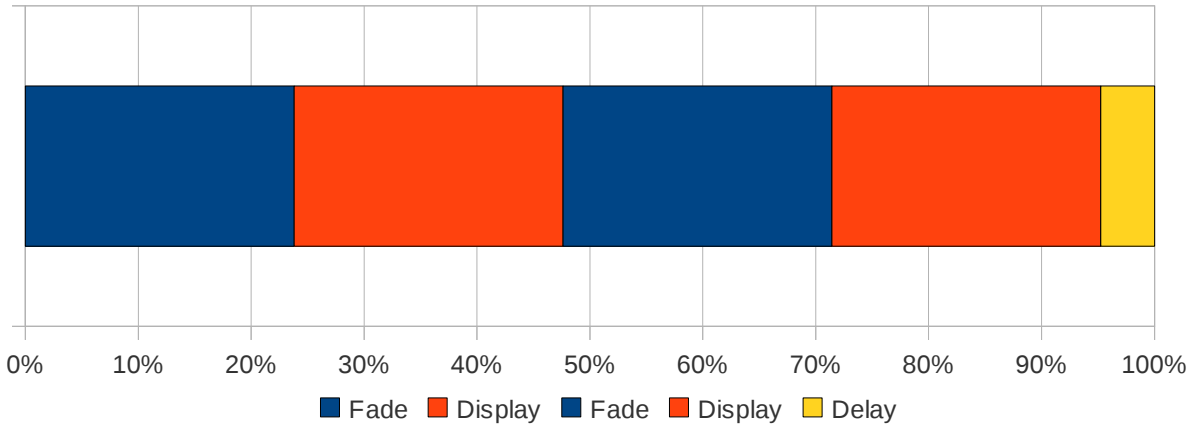

Figure 5.2: Timeline for a scene with two images

prototype the fade-in time of each image is fixed to 1000ms, so sentences with a duration less than 2000ms are not well represented by more than one image.

Image Transition

As mentioned two succeeding images are just blended over in a constant time. This is done because currently there is no information about the relation of two images or the speed in the story.
Other possibilities would be

- sliding the next image in from one side
- variating the time for the transition depending on the story
- blacking out the screen before displaying a new image in order to represent a hard cut

but to work correctly these methods need information about the spatial relationship of the images to each other and inside the story. A hard cut for instance only makes sense when the location in the story completely changes during a small timeframe, while a new image should be slided in along the axis of the movement in the scene.

Image Display

In the current prototype the images are displayed statically. The reason for this is also the lack of information from the image. An effect like zoom in/zoom out would require knowledge of the point of intrest of the image which is currently not the case. Furthermore the timeframe of each image is equal if there is more than one image associated to a scene. This is due to the lack of timing hints from the text apart from the length of the whole sentence, which is not fine grained enough in this case.

6 Summary

In this chapter the results of the presented method are discussed. First the current shortcomings are pointed out, then a possible application is discussed and last some directions which can be taken in future are presented.

6.1 Discussion

The method presented in this work is a first attempt at creating films from textual description. It explores the possibility of generating films using a huge database of annotated images. While the current results are actually more amusing than useful, several extension points were identified which should improve the quality of the results.
Furthermore improving the selection algorithm by combining remote and local filtering shows potential. On the other hand the interface for manually selecting the images can be extended to provide more aid as well. Yet there might be a usable application for the prototype already.

6.2 Possible Application

Currently digital cameras a very widespread which is not only observable on web sites like flickr but also by the increasing amount of images which users store on their computers.
To this end database based applications are used which also employ tagging in order to cope with the large amount of data. Yet it is still hard to find a sensible set of images to visualise an event like a holiday. Here the method presented in this work could help by first asking the user for a textual summary of the holiday where the user can emphasise the events important to him. After this we have the required initial position and can help the user to create a photomatic of the holiday which he then can use for sharing and presenting.

6.3 Current bottlenecks and future work

The main problem identified when developing the prototype is a general lack of information. This includes both the lack of information about text as well as the lack of information about the images. This holds back more sophisticated automated approaches. So the logical steps for future work would be:

Deducting information from text

One of the main limitations in the current work is the amount of information deducted from the textual representations. Currently only the sentence structure and part of speech is extracted, but no links between sentences are created. Therefore it is not possible to visualise sentences witch just refer to other sentences, like "she goes there". It should be possible to handle this cases by creating links between different parse trees, which could refine "she" as "a grey cat" from former sentences. Deducting this information would not only answer which noun "she" refers to, but also help to refine this object by grouping all information refering to in in the source text.
An improvement in this area would allow visualising dialogues which is currently not possible.
The amount of information to be gathered can also be improved: while currently only nouns are extracted due to the search interface limitation, gathering other adjectives and verbs would should useful. These can be used in the post processing steps to determine the "mood" of the story, which currently

has to be done manually. For this purpose adjectives could be used in conjunction with the local sort by colour to augment tag based queries. These queries currently do neither include colours nor adjectives in general due to their small ratio among all tags. Verbs on the other hand can be used to give additional hints to the presentation, which would allow better scene transitions like camera pan or zoom.

Deducting information from images

Another extension point which would improve presentation as well as image selection is deducting information from the images. Using real object descriptors for local sort would allow finding more specific images, when there is enough information gathered from text. For instance one could restrict the results to images only showing two persons when it is clear that the scene is only about "hansel and gretel".
Identifying their location inside the images would also allow zooming in on them during the presentation stage. Having positional knowledge of the objects in the image would also allow restricting post-processing filters to the objects, like just changing the colour of a car instead of the whole image.

Generating Scenes

An completely different method than using image providers would be generating the scenes. This would again require extracting enough information from the text and possibly extending them with heuristics to get enough information for a complete scene. The objects themselves could be generated from PCA data like generated by Alexandru Balan and Michael Black[15]. They use the 3D recorded data of people transformed into the eigenspace in order to perform tracking.

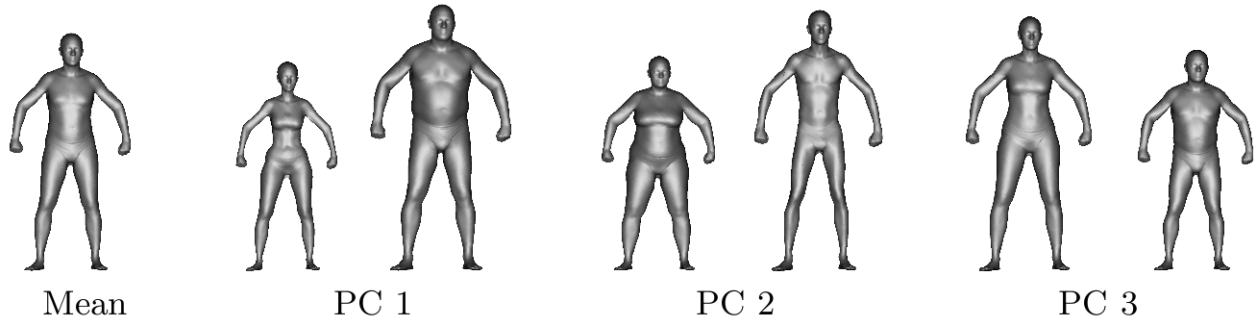

Figure 6.1: Mean 3D Body Model along with the first three PC derivations (from Alexandru Balan and Michael Black [15])

But looking at the first three principal components (Figure 6.1) one may also associate the meaning to them. For instance the first principal component(PC) can be associated with the gender, the second PC to the weight of the person and the third PC to the height of the person.
Using information from the text one could then generate actors on demand in this space.
This of course would have to be complemented with a background. For this purpose an image provider could still be used which would result in a combined method.

Generating Films

As already mentioned the process discussed in this work is general enough to produce real films instead of just photomatics. The easiest way to do this would be to use an video provider instead of an scene provider for scene retrieval. One possibility would be using youtube. The following steps could be easily adapted, since using films instead of still images basically just means using a higher resolution for the time axis and does not affect the fundamental scene representation. Therefore the options presented for post processing are still valid. But calculating the scene length would need adjustments, since it now would also depend on the film clip length. For presenting such film clips to the user for selection

techniques from film summaries could be used.
In order to further enhance the image selection step, similar images could be grouped by using the gist descriptor in order to reduce the mental requirement on the user. In could also make sense to allow combining several retrieved images into one by offering a similar interface like described in scene completion.

7 Prototype Architecture

In order to allow comparing several possible solutions there is a breakpoint introduced after the image selection stage (Figure 7.1). It is possible to save the results of image selection at this point.

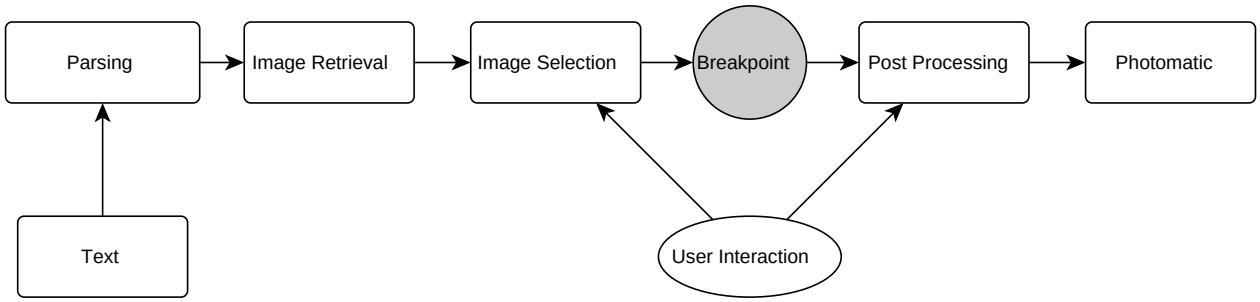

Figure 7.1: System Overview

This split also marks the split of the system in two parts; retrieval and presentation. The left part is automated up to the manual image selection part. The right part starts with the manual selection of a post-processing style and is automated from there on. The right part can be seen as a presenter for the selected results.

7.1 Retriever

The retrieval stage starts by the invocation of the stanford parser, which then tokenises the text using the Penn Tree Bank Tokenizer (Class PTBTokeniser), which follows as the name indicates the Penn Treebank tokenizing conventions. Then the keywords are extracted using regular expressions out of which the queries to flickr or optionally google images are generated. The results of the queries are then cached in order to speed up repeated execution. Then thumbnails are fetched for the results, which are displayed next to the source sentence. The full size images are only fetched on demand. The retrieval program ends with the first part of the image adaptation step; the image selection. The selection is then saved for further processing.

7.2 Presenter

The presenter begins with the selection of a post-processing effect and a voice witch allows defining a style of the story. The deferral of post-processing to the visualisation step also allows easy comparison between several effect / voice combinations. The text is composed using the festival speech synthesis runtime and all images are processed using the selected effect. The post-processing happens non destructive by saving the results in a temporary location, so in subsequent calls different effects can be selected. After that the results are presented as a Photomatic.

Effect Composition

The post processing Effects are represented by a GEGL Graph (optionally a OpenRaster graph). A GEGL Graph is basically a scenegraph where the nodes represent primitive processing steps (Figure 7.2).

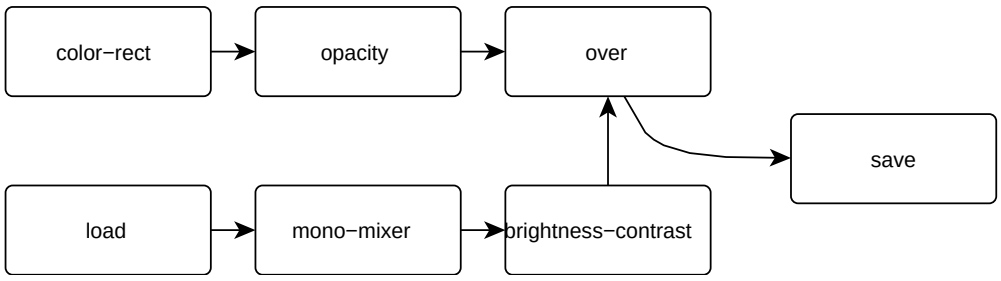

Figure 7.2: GEGL Graph used for the sepia effect

A Appendix

A.1 System Environment

As the development environment linux was chosen, because most of the necessary libraries are primary developed for linux. Furthermore linux allows easy dependency handling and installation by a package manager. But basically all used libraries are also available on windows and so the prototype should work on all platforms. As the programming language python was chosen, since it allows rapid application development and supports abstract expression of concepts by supporting a wide range of programming techniques like functional programming and object oriented programming. Furthermore the work did not include writing any performance critical parts, so the usual performance penalty coming with python is neglectible.

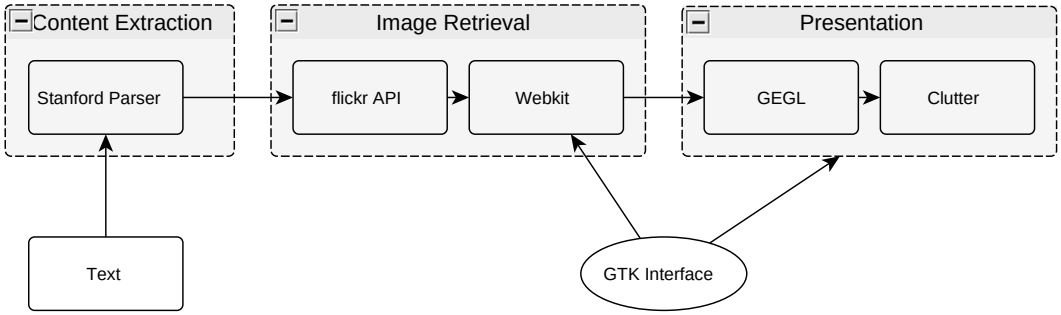

Figure A.1: Overview of the used Libraries

Stanford Parser

The Stanford parser is used for tokenizing and parsing the text. It is written in Java and no easy access from python is possible. Therefore the command line interface is used.

GTK

GTK is used as the Widget toolkit to display the interface to the user. It was chosen because I a most familiar with it and because there is already integration of the other used libraries with it. There are Python bindings for GTK.

Webkit

Webkit is used for displaying and retrieving the queried images. Furthermore the selection interface is written using javascript and also runs inside Webkit runtime. There are Python Bindings for the WebkitGTK API.

GEGL

GEGL is used for image processing. Although it is basically possible to use the serialization format for processing graph compositing, it is still unstable. Therefore the graphs are constructed directly using the GEGL python bindings.

Festival

Festival is used for speech synthesis. It is written in C++, but no real Python bindings exist. Though there are several approaches to send commands formatted in the Festival LISP dialect to a synthesizing daemon. But using festival in daemon mode is discouraged due to security flaws. Therefore the CLI text2wav interface is used.

Clutter

The Clutter animation Framework is used for Animations. It provides a scene graph like API and has python bindings.

GStramer

The GStreamer Media Framework is used for playing back the synthesised speech. Basically it is overkill for this task, but it is possible to use it in future to display video sequences as well and there are python bindings.

Installation

The used linux distribution is Ubuntu 9.04. Since at the time writing there were no release of the official release of the python bindings for GEGL, I packaged them in my PPA[1]. When added all necessary dependencies can be installed using this command:

```
apt-get install python-gegl python-webkit python-gtk2 python-clutter python-gstreamer
 festival sun-java-6-jre
```

Festival

Furthermore Festival needs some configuration, since the default installation does not ship HMM based voices[2]. In order to install the used voices download the following packages and extract them. Then copy the contents of lib/voices/us/ to /usr/share/festival/voices/us/

- `http://hts.sp.nitech.ac.jp/archives/2.1/festvox_nitech_us_slt_arctic_hts-2.1.tar.bz2`
- `http://hts.sp.nitech.ac.jp/archives/2.1/festvox_nitech_us_awb_arctic_hts-2.1.tar.bz2`
- `http://hts.sp.nitech.ac.jp/archives/2.1/festvox_nitech_us_rms_arctic_hts-2.1.tar.bz2`

A.2 Numbers and results

Precision of the image providers for the test queries

Query: animals woods trail breadcrumbs

Number of images	flickr tag mode	flickr text mode	google images
1	1	0	0
5	0,8	0,4	0,2
10	0,8	0,4	0,1
15	0,73	0,53	0,07
20	0,65	0,5	0,05
30	0,7	0,37	0,07

[1] `https://edge.launchpad.net/~madman2k/+archive/ppa`
[2] `https://bugs.edge.launchpad.net/ubuntu/+source/festival/+bug/383157`

Query: starvation woodcutter wife

Number of images	flickr tag mode	flickr text mode	google images
1	0	0	0
5	0	0	0,2
10	0	0,1	0,1
15	0,07	0,07	0,2
20	0,05	0,05	0,15
30	0,07	0,03	0,13

Query: children forest

Number of images	flickr tag mode	flickr text mode	google images
1	1	0	1
5	0,6	0,2	0,6
10	0,5	0,1	0,6
15	0,47	0,07	0,47
20	0,45	0,2	0,4
30	0,4	0,17	0,3

Example visualisation

Figure A.2: Beginning of the fairy "Hansel and Gretel"

Query: house gingerbread candies sugarwindows

Number of images	flickr tag mode	flickr text mode	google images
1	1	1	1
5	1	0,4	0,8
10	0,8	0,4	0,6
15	0,67	0,27	0,53
20	0,55	0,35	0,45
30	0,5	0,33	0,37

Query: wolf body stones

Number of images	flickr tag mode	flickr text mode	google images
1	1	0	0
5	0,4	0,2	0,2
10	0,3	0,2	0,2
15	0,2	0,2	0,13
20	0,25	0,2	0,1
30	0,17	0,17	0,07

Average over all queries

Number of images	flickr tag mode	flickr text mode	google images
1	0,8	0,2	0,4
5	0,56	0,24	0,4
10	0,48	0,24	0,32
15	0,43	0,23	0,28
20	0,39	0,26	0,23
30	0,37	0,21	0,19

Bibliography

[1] D. Klein and C. Manning, 2003, Accurate Unlexicalized Parsing, *Proceedings of the 41st Meeting of the Association for Computational Linguistics*, pp. 423-430

[2] D. Klein and C. Manning, 2003, Fast Exact Inference with a Factored Model for Natural Language Parsing, *Advances in Neural Information Processing Systems 15 (NIPS 2002)*, pp. 3-10

[3] D. Goldman et al., 2006, Schematic Storyboarding for Video Visualization and Editing, *Proceedings of ACM SIGGRAPH 2006, Vol. 25, No. 3*, pp. 862-871

[4] J. Assa et al., 2005, Action Synopsis: Pose Selection and Illustration, *Proceedings of ACM SIGGRAPH 2005*, pp. 667 - 676

[5] L. Teodsio and W. Bender, 2005, Salient Stills, *ACM Transactions on Multimedia Computing, Communications and Applications, Vol. 1, No. 1*, pp. 16–36.

[6] Y. Pritch et al., 2008, Nonchronological Video Synopsis and Indexing, *IEEE Trans. PAMI, Vol 30, No 11*, pp. 1971-1984.

[7] J. Hays and A. Efros, 2007, Scene Completion Using Millions of Photographs, *ACM Transactions on Graphics (SIGGRAPH 2007), vol. 26, No. 3.*

[8] A. Olivia and A. Torralba, 2006, Building the gist of a scene: the role of global image features in recognition, *Visual Perception, Progress in Brain Research, vol. 155.*

[9] A. Girgensohn, 2003, A fast layout algorithm for visual video summaries, *Proceedings of the 2003 International Conference on Multimedia and Expo, Vol 1,* pp. 77 - 80

[10] D. Jurafsky and J. Martin, Speech and Language Proceessing. 2nd ed., *Prentice Hall*

[11] E. Blankinship et al., 2004, Closed caption, open source, *BT Technology Journal*

[12] Eugene Charniak, 2001, A maximum-entropy-inspired parser, *NAACL 1*, pp. 132–139

[13] Wikipedia, 26. December 2008, `http://en.wikipedia.org/wiki/Photomatic`

[14] Heiga Zen et al. 2007, The HMM-based speech synthesis system version 2.0, *Proc. of ISCA SSW6*, pp.294-299

[15] Alexandru Balan and Michael Black, 2008, The Naked Truth: Estimating Body Shape Under Clothing, *ECCV*

[16] E.H. Adelson and J.R. Bergen, 1991, The plenoptic function and the elements of early vision, *Computation Models of Visual Processing, MIT Press, Cambridge*, pp. 3–20.

[17] B. Gooch and A. Gooch, 2001, Non-Photorealist Rendering, *Peters, Wellesley*, pp. 120-123

[18] G. A. Miller, 1956, The Magical Number Seven, Plus or Minus Two: Some Limits on our Capacity for Processing Information, *Psychological Review*, 63, 81-97