



Centrum voor Wiskunde en Informatica

REPORT*RAPPORT*

INS

Information Systems



Information Systems

Creating harmonious and legible colour schemes in the automated generation of multimedia presentations

Amit Manniesing

REPORT INS-E0304 NOVEMBER 14, 2003

CWI is the National Research Institute for Mathematics and Computer Science. It is sponsored by the Netherlands Organization for Scientific Research (NWO).

CWI is a founding member of ERCIM, the European Research Consortium for Informatics and Mathematics.

CWI's research has a theme-oriented structure and is grouped into four clusters. Listed below are the names of the clusters and in parentheses their acronyms.

Probability, Networks and Algorithms (PNA)

Software Engineering (SEN)

Modelling, Analysis and Simulation (MAS)

Information Systems (INS)

Copyright © 2003, Stichting Centrum voor Wiskunde en Informatica

P.O. Box 94079, 1090 GB Amsterdam (NL)

Kruislaan 413, 1098 SJ Amsterdam (NL)

Telephone +31 20 592 9333

Telefax +31 20 592 4199

ISSN 1386-3681

Creating harmonious and legible colour schemes in the automated generation of multimedia presentations

ABSTRACT

Due to the growing amount of information on the web, the user specific requirements and different characteristics of output devices, the opportunities for the automatic generation of multimedia presentations grow. The multimedia presentation generator, that resides in a dynamic environment, such as a museum website, where the user requirements, presentation device characteristics, presentation content and the domain characteristics are not known in advance, needs to be able to compose a presentation. With respect to stylistic design, graphic designers can create a template providing stylistic aspects, but as soon as any of the dynamic attributes of the system change this can result in the need for redesign. We argue that with the correct balance between form and function and by using relevant aspects of design theory, the automatic presentation generator can keep harmony and legibility factors in balance. The aim of this work is to demonstrate this approach on the example of automatic colour design with the use of colour theory from Itten and Tufte. By taking required legibility factors into account, harmonious and well balanced colour schemes, adapted to the requirements of the user, the characteristics of the presentation platform, the content's domain and discourse model can be created. We apply our approach to the domain of presentation environments for musea for fine arts.

1998 ACM Computing Classification System: H5.4, Architectures, Navigation, User issues; I7.2 Hypertext/hypermedia, Multi/mixed media.

Keywords and Phrases: multimedia semantics; automatic colour design; colour harmonisation; style-driven multimedia presentation generation

Note: This work forms the basis of report INS-R0303.

Creating harmonious and legible colour schemes in the automated generation of multimedia presentations

Amit Sharwan Kumar Manniesing

CWI

P.O. Box 94079, 1090 GB Amsterdam, The Netherlands

ABSTRACT

Due to the growing amount of information on the web, the user specific requirements and different characteristics of output devices, the opportunities for the automatic generation of multimedia presentations grow. The multimedia presentation generator, that resides in a dynamic environment, such as a museum website, where the user requirements, presentation device characteristics, presentation content and the domain characteristics are not known in advance, needs to be able to compose a presentation. With respect to stylistic design, graphic designers can create a template providing stylistic aspects, but as soon as any of the dynamic attributes of the system change this can result in the need for redesign. We argue that with the correct balance between form and function and by using relevant aspects of design theory, the automatic presentation generator can keep harmony and legibility factors in balance. The aim of this work is to demonstrate this approach on the example of automatic colour design with the use of colour theory from Itten and Tufte. By taking required legibility factors into account, harmonious and well balanced colour schemes, adapted to the requirements of the user, the characteristics of the presentation platform, the content's domain and discourse model can be created. We apply our approach to the domain of presentation environments for musea for fine arts.

1998 ACM Computing Classification System: H.5.4, I.7.2

Keywords and Phrases: Multimedia semantics, automatic colour design, colour harmonisation, style-driven multimedia presentation generation

Note: This work was carried out under the ToKeN2000/CHIME project

Chapter 1

Introduction

1. INTRODUCTION

This Master's thesis was written within the scope of the Master's program of the section Information Systems Design of the Department of Information Systems Algorithms (ISA) of the Faculty of Information Technology and Systems (ITS) at the Technical University of Delft.

The aim of this graduation work is to allow the student to work independently on a project with typical engineering aspects, such as analysis, design, realisation and implementation of algorithms, systems, methods and techniques. The research for this master project has taken place at the Multimedia and Human-Computer Interaction theme, (INS-2) of the *Centrum voor Wiskunde en Informatica* (CWI), the National Research Institute for Mathematics and Computer Science. The work concentrates on colour design issues as part of the automated generation process of multimedia presentations.

2. PROBLEM STATEMENT

In one "high information density sentence", we formulate the problem of our research:

Convey, using the presentation-oriented design knowledge of an automated informative hypermedia presentation generator, the underlying information in a variety of circumstances, taking on the one hand the delivery context and on the other hand the inherent style attributes of constituent media items into account.

By using **presentation-oriented design knowledge** we intend to include the stylistic, aesthetic and semantic aspects of a presentation, keeping in mind the balance between form and function where emphasis on function or form is the leading thread for the design process of the presentation. The derivation of the design knowledge itself is also part of this research. This design knowledge is used to guide the **automated informative hypermedia presentation generator** in creating a well-balanced hypermedia presentation. This hypermedia presentation takes **delivery context** into account, with respect to the platform and user characteristics, and on the other hand the **inherent style attributes of the constituent media items**, with respect to the domain, user and platform characteristics.

3. PROJECT PLANNING

The project is split up in five sequential phases, which to some extent can also be conducted in parallel

1. Based upon a literature study in the area of graphic design and interactive media design, the candidate will need to select a number of relevant design topics and model potentially implicit design knowledge explicitly.
2. Based upon a literature study in the area of Semantic Web languages and tools, the candidate will need to select appropriate tools for representing and manipulating the knowledge captured in phase 1.
3. The candidate will incorporate the knowledge and tools into the Cuypers multimedia transformation research prototype in order to perform practical experiments of automatically applied design rules.
4. The functionality of the design rules used for the generation of presentations needs to be evaluated against the intended semantics, the given delivery contexts, and the style attributes of the media content.
5. The research results of the previous 4 phases will be documented in a Master's thesis. The thesis should be of sufficiently high quality to be published as a CWI technical report.

4. SCOPE AND CONTRIBUTION

Graphical design issues in the automation of presentation generation are a broad topic. It comprises not only colour design issues, but also the design of spatial and temporal layout, typeface issues, such as font type, size and face, and perhaps the most important of all, cognition and perception factors. With the use of colour theory from Itten [21] and Tufte [50, 51] we will create, by taking the required cognition and perception factors into account, harmonious and well-balanced colour schemes, adapted to the requirements of the user, the characteristics of the presentation platform, the content's domain and the discourse model. These colour schemes are to be used in an automatic presentation generator prototype, which resides in a dynamic environment. Hence, the aim must be that the established structures and mechanisms facilitate content generation that generate human and machine accessible content.

5. OUTLINE

The remainder of this master's thesis is as follows:

Chapter 2 describes the underlying theory with respect to documents when dealing with automated hypermedia presentation generation, providing descriptions about hypermedia, web content evolution and the separation of style and content.

Chapter 3 explains the architecture of the prototype system Cuypers, an automated presentation generator. An example scenario describes the role of the Design Module within this architecture, the focus of our work, together with its current functionality. In this chapter it is also described how the balance between form and function is used by the Design Module in its decision process.

Chapter 4 is the result of the derivation of colour theory in general (perception and cognition) and in particular the design knowledge, with respect to Itten's equilibrium theory and Tufte's minimal effective difference theory. The results provide general rules for using colours in combination with each other, dealing with aesthetics, and the use of colours in combination with legibility problems when text is involved.

Chapter 5 provides an example scenario in which the different steps of the colour selection process will be described. A proposed architecture guides the reader in understanding the decisions made by the system.

Chapter 6 gives a description of the realisation of the Design Module, in which the example scenario will be used to make clear what happens at each step of the colour design process. The different components of the architecture will be described in detail as well as the integration of the Design Module within the Cuypers system.

Chapter 7 provides the reader with a summary and an evaluation of the project. The chapter concludes with recommendations for future work.

Chapter 2

Automatic Generation of Hypermedia Presentations

The automatic generation of multimedia presentations has been a focus of multimedia research for over a decade. The aim is to establish generation mechanisms with adaptive [9] or adaptable qualities [42] that adjust the multimedia presentation to the specific context of an individual user. Various attempts to explore and develop innovative presentation techniques have been described with respect to fully automated multimedia authoring [2], constructive text theory [3, 22], the use of relational grammar [63], cross media adaptation [8], and the use of an evolving collection of media items [14]. This section conveys the reader with a better understanding of automatically generating hypermedia presentations by describing relevant aspects of hypermedia documents, stages of evolution of web content and the resulting separation of content and style.

1. HYPERMEDIA DOCUMENTS

Hypermedia documents can be seen as a combination of hypertext and multimedia documents. Table 2.1 from [54] provides a general classification.

	Static media	Time-based media
Linear structure	Text	Multimedia
Non-Linear structure	Hypertext	Hypermedia

Table 2.1: Classification of electronic documents

Hypermedia presentations consist, thus, of the combination of multiple hypermedia elements, connected through temporal synchronization. The presentation itself is embodied in a document in which the temporal and spatial relations are encoded. This document can be for example, represented in the languages SMIL 1.0 [59], SMIL 2.0 [61] or HTML+TIME [43]. Another document type to embody a hypermedia presentation is the Flash format from Macromedia. This document type, however, is only human readable with respect to content, in contrast to SMIL and HTML+TIME where the content is human *and* machine accessible. Due to the lack of machine accessibility we will not consider Flash in this work.

The automatic generation of hypermedia presentations allows authoring at levels of abstraction higher than the final presentation [40], as well as the ability to reuse user, network and platform adapted content. To achieve this goal, an essential requirement for the auto-

matic generation process is to separate *content* from *style* [57]. In the next sections we show how this requirement developed over the years in the best known dynamic environment, the www.

2. WEB CONTENT EVOLUTION

With the introduction of the first graphical web browser, Mosaic, the applicability of hypermedia, i.e. documents combining text, links, images, sounds and video, became popular. In the first year after the introduction of Mosaic, the number of web servers grew exponentially from 100 to 7000 [49]. Over the years we saw three distinct steps in the development of Web content [55].

First generation Web content is known for its plain hypertext documents, which often were handwritten. Hypertext was considered to be the glue of the WWW [53]. The representation language of hypertext then was HTML.

When content or style needed to be changed, this type of generation of documents proved its inflexibility because of its handmade characteristics. Especially when the content was located in a database or was subject to frequent updates. This resulted in the development of the next generation of web content tools.

Second generation Web content is the current state of the art of Web content. It is much more flexible than its predecessor because of a range of new technologies based on the automatic generation of HTML content. An example of the use of second generation Web content can be seen on almost every e-commerce website. The presented data often resides in a underlying database, where it is extracted based on user-queries and then automatically arranged and displayed using XSL Transform (XSLT) (a language for transforming XML documents) and Cascading StyleSheets (CSS)¹. The advantage of this technology is the separation of content and style that for example allows reuse of content on different output devices.

Musea, where visitors can be provided with a visit to virtual exhibitions, collections and galleries in the form of a multimedia presentation, are good examples of environments for second generation Web content as well. Over the last decade musea have digitised their collections, to provide access to the general public, see also [31]. These visually pleasing, often handmade virtual collections, galleries and presentations lack, however, adaptable qualities, which can provide a better fit to the user's requirements. Attempts to create user-adapted multimedia presentations resulted in the automatic generation of hypermedia presentations. These types of presentations supply opportunities for third generation web content.

Third generation Web content aims at the presentation environments, where the content is *not only* human but also machine processable. The "Semantic Web", as envisioned in [7], is a vital ingredient for the next generation Web content. Currently the Semantic Web² is based on descriptive languages such as Resource Description Framework (RDF), RDF Schema (RDFS) and Web Ontology Language (OWL) that allow automatic reasoning on web content.

3. CONTENT, STYLE AND STRUCTURE

In both previous sections we referred to the need to separate content from style, by using XML-based languages to capture the content and let CSS deal with the style issues. Dealing with presentations, however, requires a third essential ingredient, namely the presentation structure, see also [57]. When the presentation structure is similar to the content structure

¹A typical example are the websites of the affiliated companies from the Boekhandels Groep Nederland (BGN), for example the website of Broese*) at <http://www.broese.net>.

²<http://www.w3c.org/2001/sw/>

the two ingredients content and style suffice. Presentations, however, often use a different structure from the content structure, resulting in a transformation step, such as provided by XSLT, which is needed to transform the content structure into the correct presentation structure. Figure 2.1 from [58] illustrates the dependencies between these three ingredients.

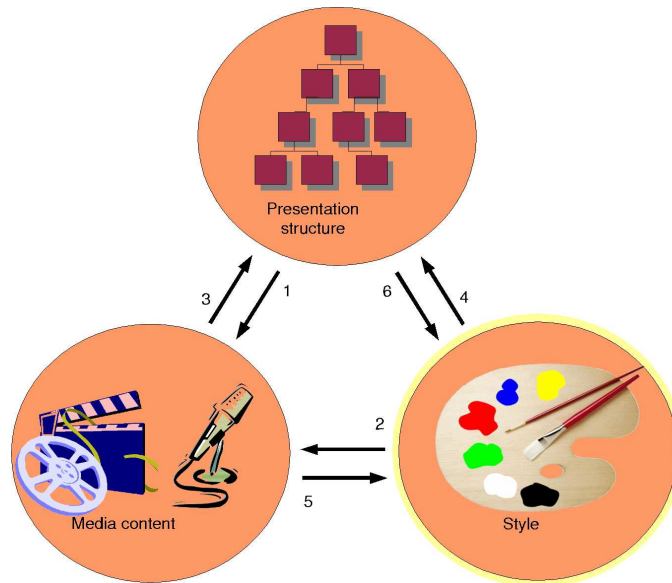


Figure 2.1: Dependencies between content, presentation structure and style

The different dependencies between presentation structure, style and content are:

1. Grouping in the presentation structure determines the selection of media items
2. The selected style determines the selection of media items
3. The grouping of the presentation structure depends on the semantic relations among media items
4. The layout aesthetics determine the grouping
5. The selected different media items determine the overall style
6. Grouping determines presentation style

As we are interested in automated colour design the relationships between style and presentation structure and between style and content are relevant. We thus have to keep in particular the relationships 2, 4, 5 and 6 in mind.

4. CONCLUSION

The brief overview of this section showed that the various dependencies between presentation structure, style and media content require a balanced application of formal and functional aspects within the automatic process of multimedia presentation generation. In the next section we investigate an environment that take these various aspects into consideration, namely Cuypers.

Chapter 3

The Cuypers system

As outlined in chapter 2, the exploration and development of innovative presentation techniques has been an important topic of research. These techniques are based upon the assumption that the material to be used and the user requirements are already known in advance, resulting in context restricted presentations. One of the few systems, if not the only one, which is situated in a dynamic environment, where neither the individual user requirements nor the requested material can be predicted in advance, has been developed by Jacco van Ossenbruggen and Joost Geurts [15, 55] within CWI's theme INS-2. The approach of a dynamic environment requires the need of knowledge, which provides a balance between media content, meaning, usability and aesthetics. As it is the Cuypers environment we integrate our work in, it is useful to explain it in more detail in the following sections.

1. INTRODUCTION

Cuypers is a research prototype system, developed to experiment with the generation of Web-based presentations as an interface to multimedia databases. In this section the architecture used in Cuypers will be outlined. The chapter concludes with an example scenario to demonstrate the Cuypers system in use. See [15, 17, 55] for a detailed description of the system.

2. CUYPERS' ARCHITECTURE

Before going through the system, its technical environment will be given some attention, see Figure 3.1. In general Cuypers distinguishes between the client and server side. At the client side a standard web client is sufficient to present the automatically generated presentation. On the server side resides the Cuypers system, the actual presentation generator. This engine is, on request, fed by an Information Retrieval system, which retrieves its information from a multimedia DataBase Management System (DBMS) with *knowledge* embedded in an annotation DBMS. At the time of writing this report, the demo of the Cuypers system works with an annotated media database from the Rijksmuseum of Amsterdam. Furthermore an *off the shelf* HTTP server (Apache) is used.

We now describe the 5 modules¹, shown in Figure 3.1.

User Module guides, with the use of rules, choices of values of system attributes, such as colour, text size, disabilities etc., to determine the most suitable user profile. Ex-

¹These modules are hardcoded at the time of writing.

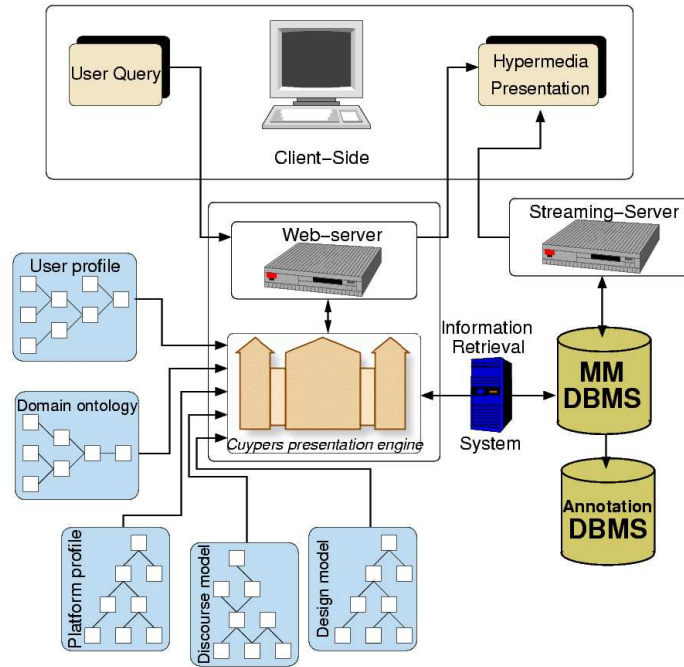


Figure 3.1: Cuypers' technical environment with its 5 modules

tensive research on the field of user modelling, see [25, 44], provides applicable knowledge for the subject of this thesis, such as vision disabilities of the user.

Platform Module provides information about the technical environment, such as network, screen and sound characteristics. Note that this implementation at the time of writing is hardcoded in the web interface.

Discourse Module is responsible for establishing relevant information for the communication structure of the presentation. One of the possibilities is giving a weight to information units, such as text or images, depending on the importance of their role for the communication goal. The discourse module is also responsible for the micro- and macro navigation within the presentation. At the time of writing research focus on ontology-driven discourse [16].

Domain Ontology supplies the system with specific domain knowledge. In our case the domain ontology includes knowledge about specific styles of art, such as the main colours of the “De Stijl” art movement, namely red, yellow and blue.

Design Module contains rules, methods and facts that make it possible to automatically design the stylistic aspects of a presentation, with respect to layout, typography and colour design. The Design Module is the module the remainder of this thesis focusses on.

The data provided by the User module and the Platform module can be encoded using Composite Capabilities/Preference Profiles (CC/PP) [60]. This is a RDF-based framework which provides platform specific data, such as hardware and software profiles, as well as user specific preferences within the output device's set of options.

The different modules produce data to guide the Cuypers Presentation Engine (CPE) in making the presentation. For ease of reusability we first introduce a scenario that describes the various steps Cuypers takes for generating a multimedia presentation. Based on this scenario we then introduce in section 4.1 examples of data that can be expected from the different modules.

3. EXAMPLE SCENARIO

This section describes the different steps taken by the CPE, where a particular user asked the system to describe the *Paintings* from the art movement “De Stijl”². The information retrieval back-end queried its annotated multimedia DBMS and retrieved four images of paintings that are annotated as being of style “De Stijl”, with the accompanying titles and year of creation. It also comes up with a textual description of the “De Stijl” movement. Looking in an abstract way at the various generation steps performed by the CPE, we can identify five steps, as outlined in Figure 3.2 from [55]. Each of these steps is described now in greater detail.

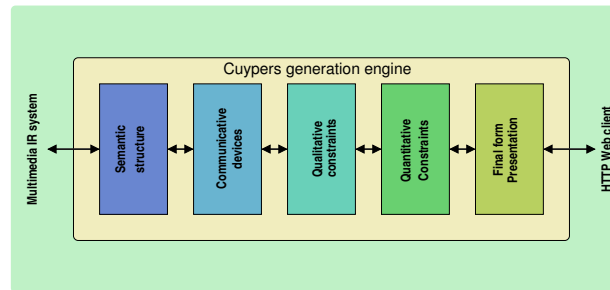


Figure 3.2: Levels of abstraction in the Cuypers generation engine

3.1 Example scenario through the levels of abstraction

The example scenario will be described in a top-down approach, starting with the semantic structure level to the final form presentation, of which an example presentation is portrayed in Figure 3.3³.

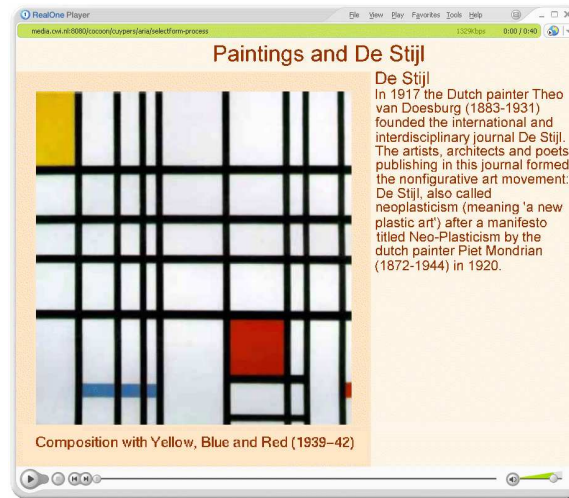


Figure 3.3: Potential SMIL 2.0 presentation in RealOne

Semantic Structure level Based on the user query, requesting the description of the paintings from the art movement “De Stijl”, a presentation is constructed. The multimedia

²A somewhat similar example is given in [15, 17, 55], where the *chiaroscuro* technique by *Rembrandt van Rijn* is used.

³The image of the painting was extracted from [10].

DBMS provides the CPE with images and text regarding the “De Stijl” movement. Figure 3.4⁴ shows the tree structure using Rhetorical Structure Theory (RST) [29]. The supplied text is an *elaboration* of the concept “De Stijl” and the images supply the *examples* of Paintings from “De Stijl”. The system decided to try and preserve the order in which the paintings were created and decided on a sequence relation. The dates of creation are supplied by the annotation DBMS. The RST nucleus/satellite relations are encoded using an XML schema, as shown in Figure 3.5.

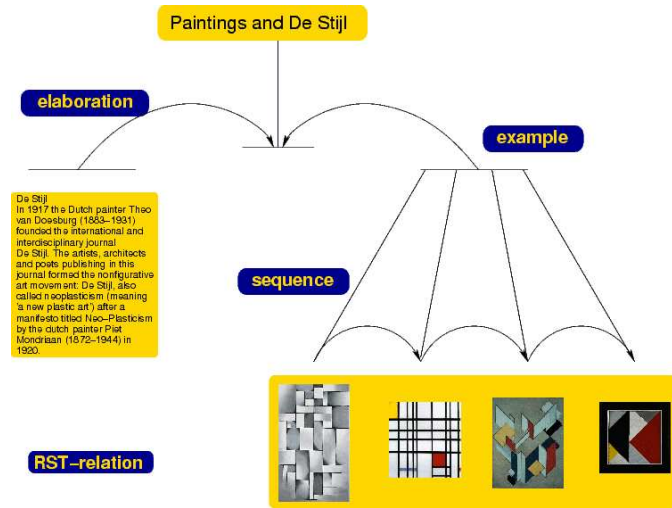


Figure 3.4: Input representation as RST tree

Note that the old Cuypers implementation did not provide automatic generation of the rhetorical structure, it was hardwired in the system’s multimedia information retrieval system. The new version also generates this structure, as described in [16].

```
<!DOCTYPE presentation PUBLIC "-//CWI/DTD Rhetorics 1.0//EN" "rhetoric.dtd">
<presentation xmlns="http://www.cwi.nl/media/ns/cuypers/rhetoric">
  <media id="title" ... refs to content/metadata database .../>
  <media id="img1" ... />
  ...
  <media id="img4" ... />
  <media id="de_stijl_paintings" ... />
  <nsRelation>
    <nucleus>
      <mediaref idref="title"/>
    </nucleus>
    <satellite type="example">
      <mnRelation type="sequence">
        <nucleus>
          <mediaref idref="img1"/>
        </nucleus>
        ...
        <nucleus>
          <mediaref idref="img4"/>
        </nucleus>
      </mnRelation>
    </satellite>
    <satellite type="elaboration">
      <mediaref idref="de_stijl_paintings"/>
    </satellite>
  </nsRelation>
</presentation>
```

Figure 3.5: XML encoding of the presentation’s rhetorical structure

⁴Images taken from [39, 10, 19].

From the semantic structure level to the final form presentation we can distinguish four levels of abstraction, namely communicative device level, qualitative constraints level, quantitative constraints level and the final form presentation level, which will be described below, (for an extensive description of these steps see [55]).

Communicative devices This step provides the spatio-temporal layout of the final form presentation based on the RST structure, combined with knowledge about the underlying domain, the user preferences and the device capabilities. Elements are grouped according to the “communicative device” they belong to, see also [41].

Qualitative constraints The communicative devices set in the previous level do not provide the system with mutual relationships among the different media. This is done in the qualitative constraint level, which converts the communicational hierarchy into a graph structure. Constraints are set to x and y coordinates and to the duration attribute. When the screen size is, for example, too small to allocate space for all the images a duration attribute which provide the relation between elements can be set to *before*, indicating that the elements are shown one after the other.

Quantitative constraints The last level before the final presentation converts the qualitative constraints to numeric, or quantitative, constraints. Here the qualitative constraints are reformulated to numerical constraints. Together with the actual sizes of the images and acceptable padding distances, the system tries to solve the constraints.

Final-form generation The processed information in the previous steps is used to generate a SMIL 1.0, SMIL 2.0 or a HTML+TIME presentation. In Figure 3.3 the potential SMIL 2.0 presentation can be seen in a player interface, namely RealOne. Figure 3.6 on the next page displays the SMIL code for this presentation. Note that this code has been adapted for readability.

This example scenario provided the decisions and steps taken during the creation of a presentation conveying information about paintings from the “De Stijl” art movement. Issues like information retrieval, presentation structure and layout are covered by the different levels. Graphic design with respect to colours, however, is not considered during the creation of this presentation. The current colours are in fact hardcoded in the presentation generator. It is the task of the Design Module, see Figure 3.1, to deal with explicit graphic design issues. The next sections provide the context in which the Design Module operates.

4. THE DESIGN MODULE

The Design Module is responsible for the design of the stylistic aspects of the presentation with respect to layout, typography and colour design. Within the abstract descriptive level of the Cuypers architecture all the abstract levels play a role, with respect to graphic design. One could say that stylistic issues are orthogonal placed in the abstract levels of Cuypers. In section 3 we showed that the selected different media items determine the overall style. We also saw that the presentation style was determined by the presentation structure, which is provided by the communicative device level. Both the qualitative and quantitative constraint levels are in combination with the communicative device level responsible for the structure of the presentation, an important element of the Design Module. Note that the spatial and temporal design decisions have already been part of research by Joost Geurts [15] and therefore will not be part of the subject in this thesis. We will concentrate on the use of colours, which is mostly dealt with, within the last level, the final-form generation. One has to keep in mind that the challenge in stylistic design is to create an equilibrium between form and function, where aesthetics and functionality both play an important role. To achieve this equilibrium the Design Module requires information from the other modules. The next section, therefore, describes the relation between the other modules and the Design Module.


```

<?xml version="1.0" encoding="iso-8859-1" standalone="yes"?>
<smil xmlns="http://www.w3.org/2001/SMIL20/Language"
      xmlns:cuypers="http://www.cwi.nl/~media/ns/cuypers#"
  <head>
    <meta content="CWI Cuypers Presentation Generator v0.8" name="generator"/>
    <meta content="www.Token2000.nl" name="project"/>
    <layout>
      <root-layout width="694" height="551" backgroundColor="#fdf5e6"/>
      <region id="title1" ...>
        <region id="title-region" ...>
          </region>
        <region id="hbox1-region" ...>
          <region id="slideshow1-region" ...>
            <region id="figure1" ...> ... </region>
            ....
          </region>
        <region id="Text-region" ...>
          </region>
        </region>
      </layout>
      <transition id="Image1-trans-in" type="fade"/>
      <transition id="Image1-trans-out" type="clockWipe" .../>
      ....
    </head>
    <body>
      <par id="root1">
        <audio src="http://../bach.allegro1.aiff" dur="40"/>
        <par id="title1">
          <text id="title" dur="40"/>
          <par id="hbox1">
            <seq id="slideshow1">
              <par id="figure1"> ... 1st painting + label ... </par>
              ....
            </seq>
            <text id="Text" region="Text-region" begin="0" src="http://.." dur="40"/>
          </par>
        </par>
      </par>
    </body>
  </smil>

```

Figure 3.6: SMIL 2.0 representation of the presentation shown in Figure 3.3

4.1 Relation between the Cuypers' modules

The Design Module is the only module that is supplied with input from *all* other modules. Some of this information was already elaborated in section 2, such as user, platform and domain characteristics together with discourse information. Keeping the “De Stijl” example scenario in mind we go a second time through the different modules:

User Module This module provides the Design Module with user specific knowledge. In our example this information might be colour-blindness of the user.

Platform Module The platform module provides platform characteristics. In our example we are using a colour screen with a resolution of 800x600 and have the ability to use sound.

Discourse Module The discourse module provides us with importance weights of the constituent media items, for example the text element is more important than the title and is thus ranked higher.

Domain ontology The domain ontology provides the domain specific characteristics. In our example this can include the use of the typeface family “De Stijl” and the use of the typical “De Stijl” colours, namely red, blue and yellow.

The information provided by the different modules constrains the Design Module and narrows its search space.

4.2 *Form Function Balance*

An automatic presentation generator has to be able to balance the functional aspects of a presentation. These need to address the information needs of the user and the presentation's aesthetic form [34, 32]. In the previous section we saw, with respect to the Design Module, the importance weights provided by the Discourse Model. These importance weights provide relevant data, which will support the Design Module deciding between form and function. Other modules, for example, the user module, which can provide the Design Module with the user's expectation and communicational goal, also aid to the decision making in the colour design process. The heuristics used by the Design Module will use all the incoming data to solve the resulting constraints of the form and function balance.

5. CONCLUSION

This section provided the underlying framework for the automatic presentation generation and explained the uniqueness of the Cuypers system due to the dynamic environment in which it operates. Its modular framework offers a way to support automatic presentation generation, based on relevant aspects, such as content, meaning, usability and aesthetics. The Design Module, which is the focus of our work, now needs to be instantiated with knowledge about stylistic graphic design. This stylistic design knowledge and its theoretical basis are the topics of the next chapter.

Chapter 4

Colours and Legibility Factors in Graphic Design

1. INTRODUCTION

While in Cuypers the temporal and spatial constraints are already implemented, other graphic design factors also need to be automated as well. One important factor is the use of colours and their combination. Everybody has some intuition about these aspects, but trying to make this implicit knowledge explicit can be difficult. In the following sections, we try to establish a foundation on which explicit rules can be created.

2. COLOURS

Making design decisions when dealing with colours is a complex process. There are, for instance, millions of colours to choose from. Moreover, every choice of colour is also based on a feeling of harmony or cultural coding. Such design decisions, partially based on aesthetics, can be made easier by *understanding* colours. This section starts, therefore, with some physics about colours and a description of how the physics shape the colour perception of humans. We use the established physical and perceptual facts as the basis for the description of computational colour models. We then describe the cognitive aspects of colour. The section on colours concludes with a base for design decisions rooted in implicit aesthetic rules, namely colour harmony schemes.

2.1 Physics

In 1665 Newton set the base for the first colour wheel [26], with the now well known *experimentum crucis*. In those days it was already known that light could be split into the colours of the rainbow by using a prism. However, people thought that the incoming light was modified somehow. Newton separated one of the colours and let this colour go through a second prism, but this light did not split up, see Figure 4.1. He concluded that the incoming light was split up in elementary colours by the prism. By recombining the colours through a reversed prism he could recreate the incoming light. His conclusion was that white light is the result of adding all visible colours. Thus he stated that colour was part of the light and not part of the prism and he concluded that white light was a mixture of refrangible rays.

Newton was a great music lover. Thus he organized the light harmoniously on a Dorian music scale and divided it into seven segments, similar to an octave that displays seven sound intervals. Figure 4.2 from [5] displays Newton's colour circle with its seven intervals, where the sizes are proportional to their respective colour's intensity in the spectrum. The

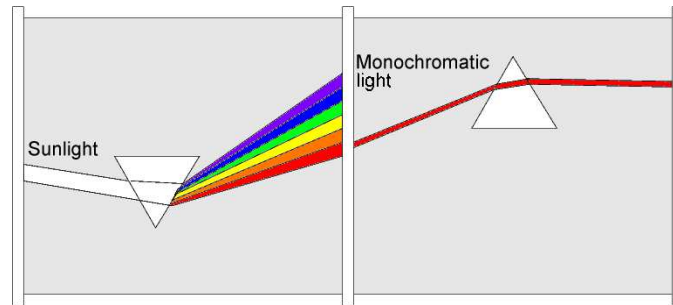


Figure 4.1: Newton's experimentum crucis

colours Newton distinguished were rubeus (red), aureus (orange), flavus (yellow), viridis (green), caeruleus (blue), indiculus (indigo) and violaceus (violet). The borders between the colour segments were given a letter according to the individual sound tones associated with the Dorian music scale.

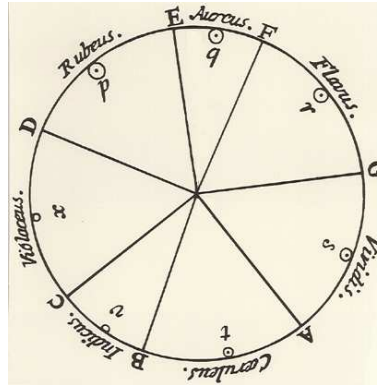


Figure 4.2: Newton's colour circle

In 1861 Maxwell showed that light was a form of electromagnetic energy which travels in waves [48]. Table 4.1 shows the colours from Newton's colour circle with their corresponding wavelengths [30]. The wavelength of electromagnetic waves is measured in very small units called Ångström, these are one tenth of a nanometer ($1 \text{ nm} = 10^{-9} \text{ m}$). Both Ångström and nanometers can be used as measurement units for the electromagnetic spectrum.

colour	latin name	wavelength
violet	violeus	390-430 nm
indigo	indicus	440-450 nm
blue	caeruleus	460-480 nm
green	viridis	490-530 nm
yellow	flavus	550-580 nm
orange	aureus	590-640 nm
red	rubeus	650-750 nm

Table 4.1: Colours and their corresponding wavelengths

A wide range from 400 nm (violet) to 750 nm (red) is the range of visible colours, see Figure 4.3 from [28].

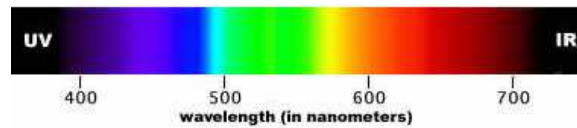


Figure 4.3: The electromagnetic visible spectrum

In 1900 Planck suggested that Maxwell's electromagnetic energy comes in discrete amounts and is thus quantized. Partially based on Planck's ideas Einstein published in 1905 a paper called "*On a heuristic viewpoint concerning the production and transformation of light.*" It showed that Maxwell's theory was unable to explain photoelectric emissions and proposed a new theory of electromagnetic radiation. Einstein stated that light is composed of photons, also called quanta, little "wave packets", carrying a fixed amount of energy. Later on Arthur Compton confirmed the existence of photons.

Having established that light exists of photons travelling in wave packets, where each wavelength stands for a certain colour, we are in the position to explain how the human eye responds to incoming light.

2.2 Colour vision

If we look at the eye, light enters through the lens and shines on the retina. On the retina there are two types of photoreceptor cells, the cones and rods. These photoreceptor cells absorb the photons and generate neural signals that initiate vision. The rods and cones have the same architecture in which photons are absorbed by a certain pigment. After absorbing these photons a chemical and electrical process takes place and the amount of recently absorbed photons is transmitted by the rod or cone to the brain.

*The chemical and electrical process*¹ The pigment molecules are located near the outer segment of the photoreceptor cells. There are roughly 100 million of these molecules in one photoreceptor cell. These molecules are members of a class of receptor molecules which sense signals coming from outside themselves. These kinds of molecules share the same architecture, where a single chain of amino acids (the protein) is embedded in a lipid membrane. In the centre of the molecule a chromophore is embedded. When a photon is absorbed by this chromophore, it rotates at one of its molecular bonds. This effect is called *cis-trans* isomerisation, and is in fact the only direct effect the absorption of light has. When the chromophore rotates, it changes the three dimensional shape of the protein, which now has catalytic properties. In this state the protein activates another protein called transducin. This catalytic process produces hundreds of these transducin proteins.

Transducins activate a third type of protein called phosphodiesterase. This enzyme breaks down a messenger substance, cyclic GMP (cGMP) inside the outer segment of the photoreceptor cells. The substance cGMP causes in the membrane of the outer segment a *trapdoor* to open in which sodium (Na^+), magnesium (Mg^{2+}) and calcium (Ca^{2+}) ions can enter. This is balanced by an outgoing stream through a sodium, calcium and potassium (K^+) exchanger at the outer segment [66]. The enzyme breaks down the cGMP, resulting in the closing of the trapdoor. This means that no calcium, sodium and magnesium ions can enter through the membrane, but that the outgoing stream continues. This finally results in a decrease of calcium ions and thus in a decrease of voltage inside the cell (hyperpolarization), causing the voltage of the cell to become more negative within the entire cell compared to the "normal" nerve cells. When we see bright light the voltage can be -70 mV whereas in darkness it typically is -30 mV. The synaptic ending of the photoreceptor cell is also influenced by the voltage shift, causing the amino acid glutamate to be less transmitted to the bipolar cells. These cells pass this information to retinal ganglion cells located on the inner surface of the retina, see Figure 4.4. From here, long fibres carry the message

¹Taken from [6].

over the optic nerve to the brain, where processing takes place.

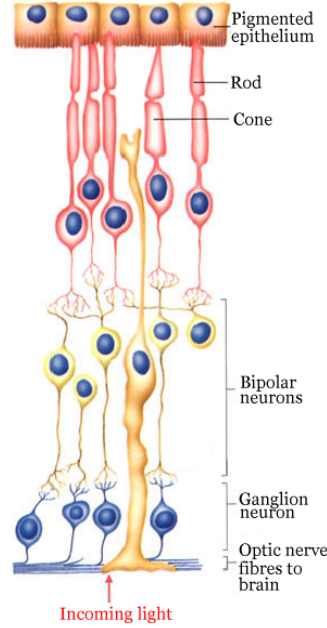


Figure 4.4: A diagrammatic cross section of the retina [28]

Sensitivity and distribution Sensitivity is inevitably related to intensity, *the amount of absorbed photons on a certain surface in a fixed amount of time*. Rods are extremely sensitive and are able to respond to a single photon. Cones are less sensitive and provide vision by ordinary daylight. These photoreceptor cells are responsible for colour vision. As described in the chemical and electrical process, the photoreceptor cells send no wavelength information, they are only transmitting the amount of photons absorbed by the cells. But why is the wavelength then of such importance?

The wavelength of the incoming light determines the probability that the photoreceptor cells absorb the photons of this light. Most human eyes have three types of cones where each has a peak sensitivity, giving us the ability to separate three different parts of the spectrum: the red, green and blue part. In table 4.2 the approximated peak sensitivity of the three types of cones is described in nanometers, see also Figure 4.5. The L, M and S stand for Long, Medium and Short, referring to the wavelength of their peak sensitivity.

name	wavelength	colour
L-cones	599 nm	red
M-cones	555 nm	green
S-cones	446 nm	blue

Table 4.2: The three type of cones and their peak sensitivity.

For the L-cones it can be seen that their peak sensitivity is 599 nm, this means that light with a wavelength of 599 nm is most likely to be absorbed by this type of cone. The perception of what colour we are experiencing is thus determined by the combination of stimulated cones and the intensity they are stimulated with.

²Values from http://cvr1.ioo.ucl.ac.uk/database/data/cmfs/ciexyz31_1.txt.

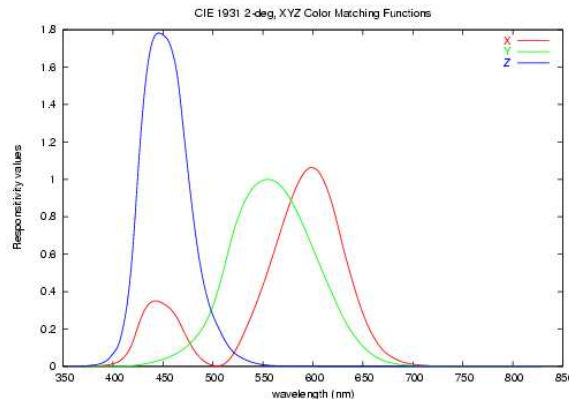


Figure 4.5: Spectral sensitivity of the CIE 1931 colour matching functions²

Figure 4.5 shows a plot of the responsivity of the CIE colour matching functions³ and can be read as the *cone sensitivity functions* [45]. These are actually linear transformations from the RGB matching functions to eliminate negative values [35].

The above theory, that states that colour vision is the result from the action of three cone receptor mechanisms with different spectral sensitivities, is called the *tristimulus theory*.

The retina contains around 120 million rods and 6 to 7 million cones. The latter are mostly concentrated in the central yellow spot. In the centre of this yellow spot lies the *fovea centralis*, a rod-free area with densely packed cones. Rods are dense elsewhere on the retina. The fovea centralis is the area where the highest visual acuity can be achieved. The 6 to 7 million cones can be divided into red cones (64%), green cones (32%) and blue cones (4%), see [36]. Green and red cones are concentrated in the fovea centralis, whereas blue cones are mostly to be found outside the fovea. Blue cones are more light sensitive than the red or green ones, however this does not compensate being outnumbered. This is reason to believe that there is some sort of a blue amplifier in the visual process of the brain.

The tristimulus theory is not only helpful in understanding colour vision, it is also helpful understanding colour blindness. There is a large percentage of people who suffer a form of colour-blindness. Because of the impact this may have on our design decisions we need to consider this.

Colour vision abnormalities Most people are trichromatics, this means they see colours using three kinds of colour sensors, namely, the red, green and blue cones as described in the tristimulus theory. There is, however, a considerably large percentage of people who are so called colour blind. This doesn't mean they can't see any colour. It means they suffer a certain abnormality in the eye's visual pigment. We can divide these colour blind people into 4 groups, namely monochromats, dichromats, anomalous trichromats and tetrachromats.

Monochromats This type of colour blindness is very rare and people with this type of colour blindness are truly colour blind and only see lightness differences. There are two types of monochromats, rod monochromats and cone monochromats. Rod monochromats have no functioning cones on their retina and cone monochromats have only one type of cones on their retina, which often is the blue one.

Dichromats The dichromats only have two types of cones instead of the normal number of three. This kind of colourblindness can be divided into three types, the protanopes,

³The method of determining these functions is explained at <http://www.adobe.com/support/techguides/color/colormodels/cie.html>.

deuteranopes and the tritanopes. The protanopes lack the L-type cone, the deuteranopes the M-type and the tritanopes the S-type. Figure 4.6 displays the electromagnetic visible spectrum perceived by trichromats and the three types of dichromats.

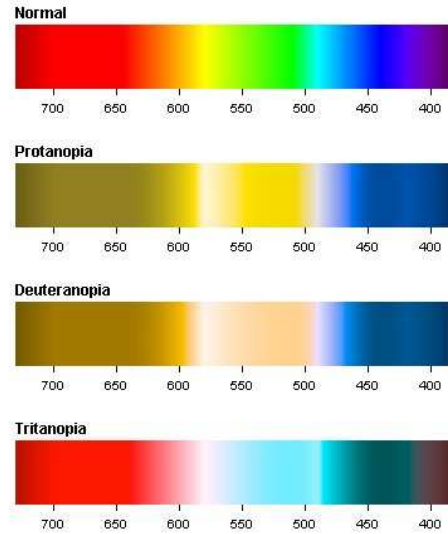


Figure 4.6: Perceived electromagnetic visible spectrum by trichromats and dichromats [64].

Missing or defective cone type	Generic defect name	Weakness
L-cones	protan	red
M-cones	deutan	green
S-cones	tritan	blue

Table 4.3: Cone defects

Anomalous trichromats People with this kind of colour deficiency have all three types of cones, but one of these types is abnormal. The abnormal cone has a different characteristic in absorbing light. This sort of abnormality varies between individuals in a range from almost dichromats till nearly normal colour vision. Here we also have three types, protanomalous trichromats, deuteranomalous trichromats and tritanomalous trichromats. The protanomalous trichromats have abnormal L-type cones, the deuteranomalous trichromats have abnormal M-type cones and the tritanomalous trichromats have abnormal S-type cones.

Tetrachromats Colour blindness is a form of colour vision *deficiency*, a very rare colour vision abnormality is *tetrachromacy*. Tetrachromats have four types of cones instead of the normal number of 3. This vision abnormality was discovered in 1948. Recent testing by Gabriele Jordan resulted in finding people having the characteristics of a tetrachromat. The testing is, however, still work in progress so Jordan still declines having found a tetrachromat. This fourth cone should be an extra cone between the L-type and M-type cone⁴.

⁴<http://www.redherring.com/mag/issue86/mag-mutant-86.html>

⁵Incidence of tritan defects has not been clearly determined.

Type of visual defect	Visual defect	Percentage of men affected	Percentage of women affected
Trichromats	None	92.002%	99.573%
Anomalous Trichromats	Protanomalous	1.08%	0.03%
	Deuteranomalous	4.63%	0.36%
	Tritanomalous ⁵	0.0%	0.00%
Dichromats	Protanopes	1.01%	0.02%
	Deuteranopes	1.27%	0.01%
	Tritanopes	0.005%	0.005%
Monochromats		0.003%	0.002%

Table 4.4: Relative abundance of visual chromatic defects

Table 4.4 portrays the three types of colour vision deficiencies. It also shows that colour vision defects affect more men than women. If we look at the genes it becomes obvious. On the X chromosome the genes for the red and green photopigments are adjacent to each other. The genes for the blue photopigments are on a different chromosome. Women have 2 X chromosomes, so they have 2 sets of green and red photopigment genes. Men have only one X chromosome, created by mixing the X chromosomes from the mother and father. Because the genes of the red and green photopigments are next to each other they sometimes mix. This is normal. Sometimes, however, the mixing goes wrong and the result can be a “defective” X chromosome. This “defective” X chromosome can result in the lack of the green or the red photopigment gene, two slightly different red photopigment genes or two slightly different green photopigment genes. If the egg contains this “defective” X chromosome and is a male embryo, then this male will be colour blind⁶.

Having described the mechanism in colour vision, we are now able to model colours in a way that a machine can work with colours. Thus, the next section addresses various *colour models* developed over time.

2.3 Colour models

Colour models are used to classify and describe colours in terms of hue, saturation, lightness, value or brightness in a machine accessible representation. In this section the most common models are described, namely the RGB, CMYK, CIE, YIQ and HSL colour models.

RGB The RGB colour model is a so called additive colour system. This system is based on the emittance of light. We start with black and by adding more colours we eventually get white. The RGB model is the most basic, and probably best known, colour system. The most important property of the RGB model is that it closely relates to the way we perceive colour with the red, green and blue cones on our retina. Most Cathode Ray Tube (CRT) displays (e.g. televisions and computer monitors) work with the RGB model and this model is also used for web graphics.

Familiar aspects of this system are primary and secondary colours.

Primary colours Primary colours are the fundamental colours of a colour system. In the RGB model this colours are red, green and blue. These colours cannot be created by mixing other colours. All other colours created in the colour model consist of a mixture of these primaries.

Secondary colours Secondary colours are created by equally mixing two primaries. This

⁶One could imagine the tetrachromats getting their abnormal colour vision, by having two X chromosomes, with one “defective” X chromosome and one normal. This means only women can be tetrachromats and the “fourth” cone could be one with a slightly different green or slightly different red photopigment gene.

means that there are three secondaries. In this case these colours are yellow, cyan and magenta.

In Figure 4.7 the primary and secondary colours are marked with the first letter of their names. In the same Figure we can also see the starting colour black. With the combination of Newton's experimentum crucis and the definition of the primary colours above, we should be able to explain why the equal mixture of the three primaries result in white.

A 3D representation of the RGB colour model is called the colour space, see Figure 4.8. On the axis of this colour space the primary colours are projected and in the origin we can see the colour black. When coding colours with the RGB model there are different methods, the most commonly used represents the colour by 3 values, each on a scale from 0 to 255 representing the amount of Red, Green and Blue. Another method is often used on web pages and is somewhat similar. Here the values are, however, hexadecimal, concatenated and preceded by a # symbol. Black can be presented as 0 0 0, or as #000000, whereas white is 255 255 255 or #FFFFFF. The representation of colour, say yellow is 255 255 0 or #FFFF00.

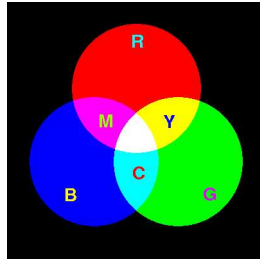


Figure 4.7: The RGB primary and secondary colours

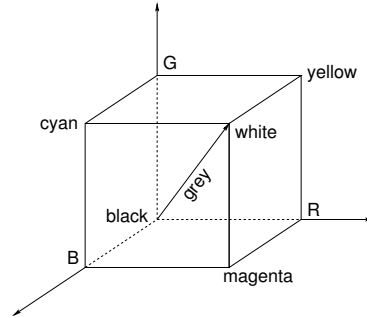


Figure 4.8: The RGB colour space

CMY(K) For printing, painting and drawing we use a totally different colour system, because here we don't see emitted, but reflected light. The object which reflects the incoming light absorbs all wavelengths except the one we see. In contrast to the RGB system, this system is called a subtractive colour system and consists of the primary colours yellow, cyan and magenta. By mixing these we get the secondary colours green, red and blue. In contrast to the RGB model the CMY model starts with white and by adding more colours it eventually results in black. However, the resulting black appears not to be deep black but more a dark brownish colour, because of this and due to the difficulty of creating grey tints, the colour black is added to the system. The name CMYK is the result of these four colours: Cyan, Magenta, Yellow and black. Most people were taught the primaries to be red, yellow and blue, even in painting class. In most cases the actual used colours then were magenta, yellow and cyan.

The CMY colour space presented in Figure 4.10 has much resemblance with the RGB colour space. In fact, when using a scale from 0 to 1 for the axis, the transformation can be done with the following equation:

$$\begin{pmatrix} C \\ M \\ Y \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} - \begin{pmatrix} R \\ G \\ B \end{pmatrix}$$

CIE colour In 1931 the Commission International de L'Eclairage (CIE) developed a device independent colour system that was based on the tristimulus theory and thus on human

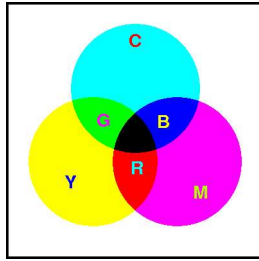


Figure 4.9: The CMY(K) primary and secondary colours

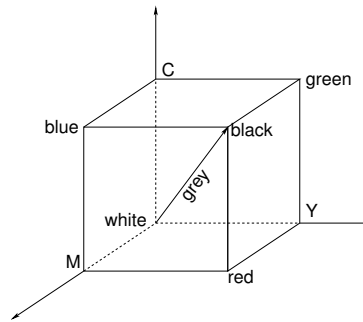


Figure 4.10: The CMY(K) colour space

perception. The differential response of the three cones is measured in three variables X, Y and Z, resulting in a 3D colour space. By projecting the Z coordinates to the X Y plane we get the CIE model, see Figure 4.11 from [38].

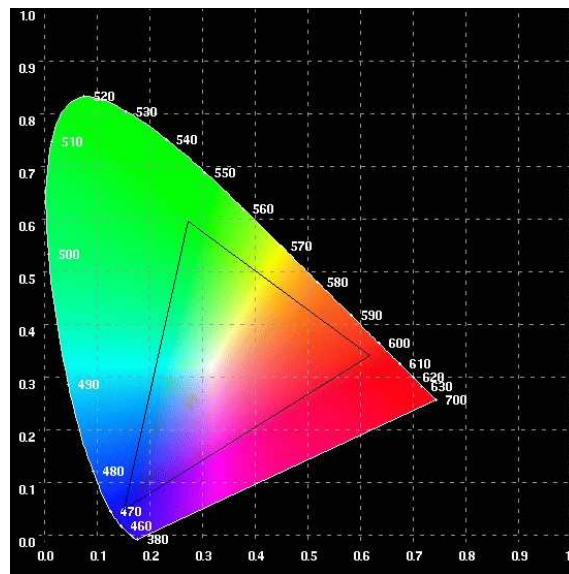


Figure 4.11: The CIE colour model with the gamut of an average RGB monitor

At the perimeter edge the pure spectral colours and wavelengths are presented. Notice the purple colours at the bottom. These colours do not have a wavelength and are created by a mixture of violet and red. By combining the different wavelengths of the spectral colours the inner part of the figure is created. White light is seen when all three cones are stimulated.

We can use the CIE model for presenting the range of colours on a *colour producer*, such as a monitor, a television set or a printer is able to display. We call this range the *gamut*. In Figure 4.11 we can see the example gamut of a typical CRT monitor represented by the triangle, which is a lot smaller than all the possible colours we can see. Because all monitors are different we have to take into account that their gamut also differs and thus that the colours we want to display are to be displayed differently.

YIQ The YIQ model is a model used for recoding RGB values for the use of televisions, in fact for NTSC standard televisions. This model is used in the US for commercial broadcasting because it is a more efficient way of transmission and it is downward com-

patible with black/white television sets. It is downward compatible because of its luminance (Y) component. This component captures our perception of the relative brightness of colours. The I and Q components describe the colour information and stand for Inphase and Quadrature. The black/white television sets thus only have to use the Y component. The conversion from RGB to YIQ is given by:

$$\begin{pmatrix} Y \\ I \\ Q \end{pmatrix} = \begin{pmatrix} 0.299 & 0.587 & 0.114 \\ 0.596 & -0.275 & -0.321 \\ 0.212 & -0.532 & 0.311 \end{pmatrix} \times \begin{pmatrix} R \\ G \\ B \end{pmatrix}$$

We will use this colour model particularly for its Y component to compare the luminance of different colours.

HSL The Hue Saturation Lightness colour system is a system that is more intuitive to artists and graphic designers. There is a variant of this system called the HSV system, where the V stands for Value.

If we look at the visible colour spectrum in Figure 4.3 on page 19, we can see that we are missing a colour: Magenta. This happens to be the mixture of wavelengths red and violet, the very ends of the colour spectrum, as seen in the CIE model. By using this knowledge we could easily create a colour circle by connecting red and violet together with magenta and filling in the inner part of the circle. Because white light is the addition of all spectral colours, the centre point is white.

A colour on the wheel can be represented by a degree between 0° and 360° , this we call the hue. Hue is a way of describing a colour in a relative way compared to its wavelength, like when we are describing colours such as yellow, red, blue etc.

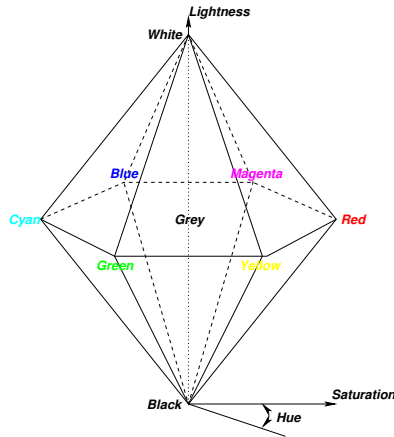


Figure 4.12: The HSL colour space

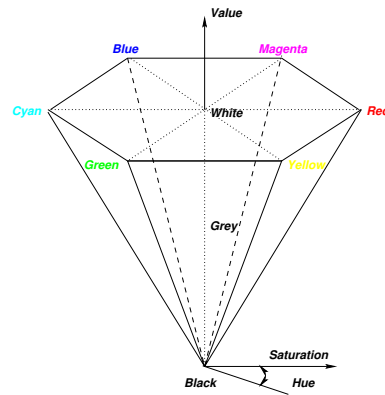


Figure 4.13: The HSV colour space

When we display the colour wheel in a 3D perspective as a dodecahedron, see Figure 4.12, where the circle is displayed as a hexagon on the X and Y axes, the Z axis can be used to display the Luminance, or the darkness/lightness of the colour. From the central axis to the border of the colour wheel there is a difference in dominance of the hue of the colour. In the centre, we have a totally desaturated colour, and at the borders we have fully saturated colours. Therefore the distance from the centre to the border is a measurement system for the saturation. This 3D projection is the HSL colour space.

Hue A value representing a certain wavelength of a colour.

Saturation The amount of hue in a colour, if the saturation is 100% we have a pure colour and at 0% there is no colour at all, and it is a grey tint.

Lightness A measure of brightness of a colour, where 0% is pure black and 100% is pure white, both regardless of the hue or saturation. At 50% there is maximum colour saturation.

In the HSV system at a value of 100% represents a maximum colour saturation and at a value of 0% there is pure black, regardless of the hue and saturation. To get white in the HSV system the saturation has to be 0% and the value has to be 100%, see also Figure 4.13. A conversion from HSL to RGB and vice versa is given in Appendix 2 and 3. Note that the transformation is not linear like the CMY \leftrightarrow RGB conversion.

One important factor of the different colour models is the ease of use. Conversion from one colour space to another is easy, so that using a model to represent the output's hardware is thus not necessary. The HSL colour space is a very intuitive colour space. We can lighten a colour by adding white, or darken it by adding black, and fully saturated colours are found on the equilateral plane. In the next sections when talking about colour models or spaces, we refer to the HSL space, except when there is explicit mention of another model.

Having introduced various colour models it is time to explain what they can be used for. We are now in the position to enter the field of colour cognition.

2.4 Colour cognition

As described in section 2.2 on colour vision, it was shown that human perception only provides the basis for the final processing of the information in the brain. This cognitive processing is a subjective task. Studies showed that certain *feelings* about colours are common, even through different cultures [65]. Symbolic values of colours, however, differ to a great extent between different cultures [20]. In this section, we aim to classify colours according to physical and psychological definitions.

Colour temperature We can make a separation between warm and cool colours. Warm colours are often associated with fire and the sun. These colours produce feelings that are warm, cosy and inviting. Cool colours, however, are often associated with ice and water and the feelings produced here range from calm and peace to sadness. When using warm and cool colours together, one is able to notice cool colours tend to recede from the viewer and warm colours tend to move forward. Using the colourwheel, see Figure 4.14, we can simply define warm colours as an clockwise interval between red-violet and yellow. The clockwise interval yellow-green to violet tends to be called cool. This definition is partially based on experiments on user experiences and described in greater detail in [21]. The intervals are however, not exactly defined because of the subjective property of colour temperature.

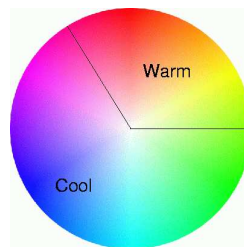


Figure 4.14: Warm and cool colours

Luminance of colours Colours can of course be separated into light and dark colours. At the very ends of light and dark colours, we have pure white and pure black. If we only stay in the white-black area, thus without using a colour, or with the use of a totally desaturated colour, we obtain all different grey tones or shades. When we use colours however, it is harder to separate the dark colours from the light colours. Everybody has some intuition

for colours, e.g. blue is dark and yellow is light, but how can this intuition be quantified? In [21] the luminance of a colour is described with the use of a figure like Figure 4.15.

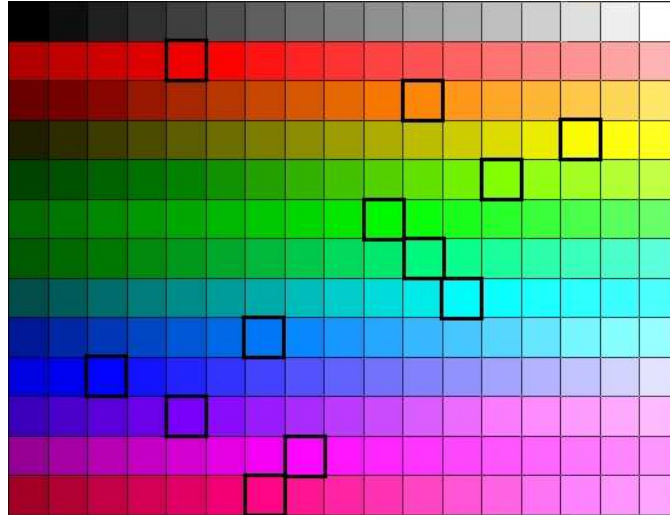


Figure 4.15: The Luminance of different hues

In this figure we see in the first row 17 equidistant steps from black to white. The other rows are the representation of 12 hues of the colour wheel with the same luminance as the greys. So every column has the same luminance⁷. These 12 hues differ 30° starting with 0° corresponding to the colour red. The trick is to spot the correct representation of the fully saturated colour belonging to its hue. In Figure 4.15 the square to the colour's nearest fullest saturation is represented by a thick border. We can see that fully saturated blue is in the third column whereas fully saturated yellow is in the 15th column. We can conclude from this, that yellow is brighter than blue. We can extrapolate this for the other colours.

This method is not very useful to define the luminance of a colour in an automated presentation generation. In section 2.3 we outlined the YIQ model for the NTSC colour television standard. In this model the Y component described the relative brightness of a colour. We will use this component to quantify the luminance of a colour. In fact, this Y component was used to create Figure 4.15.

In Figure 4.16 the Y component of the YIQ model is plotted, with the use of Matlab, against the hue and lightness value of an HSL-coded colour. When there is no saturation all colours look the same (grey) and there will be no difference between hues. With a saturation of 100% the difference between different hues will be maximal and that is the reason that this graph is plotted at a saturation level of 100%. In this figure, we can see that, when we use the maximum lightness we have the maximum brightness and vice versa. At a lightness level of 127, we have the pure spectral colours. Figure 4.17 is a slice of Figure 4.16 on the lightness value of 127. At hue value 60° there is yellow and we can see that yellow is the brightest colour, at 240° we can see that blue is the darkest colour. Note that the colours used in the plot are merely for better *readability*.

2.5 Colour Harmony

The term harmony is not only used in music, it has many areas which it applies to. In this thesis the definition from [52] is used:

A pleasing combination of elements in a whole.

Colour harmony is therefore a pleasant combination of colours. But what can be called pleasing? It is obvious we are entering the subjective area of aesthetics. In this section we

⁷on screen

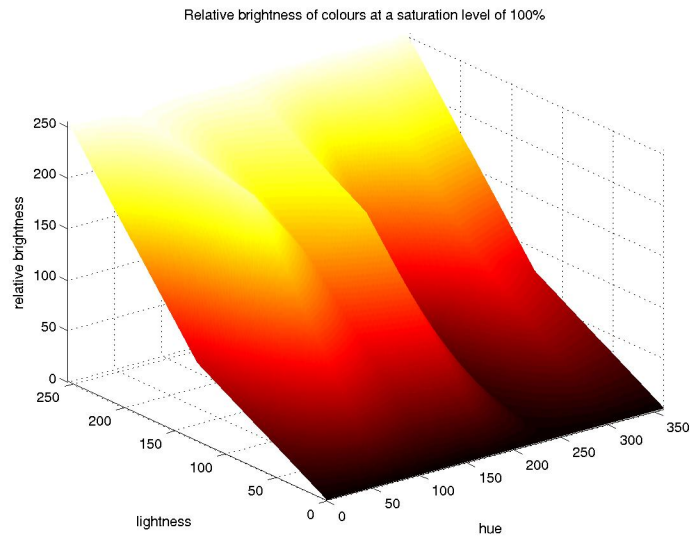


Figure 4.16: The Relative brightness of colours at a saturation level of 100%

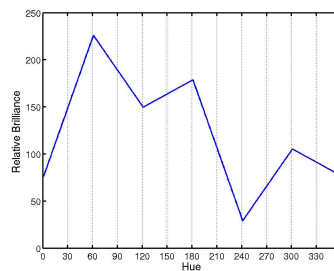


Figure 4.17: The Relative brilliance of the pure spectral colours

try to describe a way of defining harmonious colour schemes.

If a colour representation is not in harmony it can be boring or, even worse, tiring to look at. To keep the viewer's attention the colours have to be in harmony. However, to obtain attention a non-harmonious colour selection can also be applied. The harmony of colours is mostly subjective.

Two theories can help us in finding harmonized colour combinations. Both are, however, somewhat contradictory. The first one is described by Tufte in [51] and is called the minimal effective difference. The second one is described by Itten in [21] and is based on the equilibrium state of the eye.

Minimal effective difference The design strategy of the minimal effective difference is described as [51]:

*Make all visual distinctions as subtle as possible,
but still clear and effective*

When dealing with colours this means finding colours which are almost similar, with the colourwheel in mind, this means finding colours which are adjacent on the colour wheel. We can distinguish three different harmony schemes based on the minimal effective difference.

Achromatic An achromatic colour harmony scheme is a scheme which lacks all colour. We only have black, white and everything in between. In Figure 4.18, we can see how this can be projected on the colour wheel. The value of the hue in the 3d HSL colour space does not matter because our saturation used is 0. So the lightness and hue can obtain each possible value, as long as the saturation is 0.

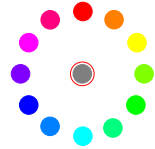


Figure 4.18: Achromatic

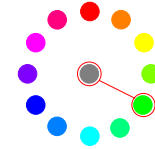


Figure 4.19: Monochromatic

Monochromatic The monochromatic colour harmony scheme is used when we use a single hue. Adding more white or black to the colour creates different colours. Reverting to the 3D HSL colour space, a monochromatic scheme implies setting the hue to one single value and allowing the saturation and lightness fluctuate between 0 and 100%. In Figure 4.21 we can see several different colours with a hue of 120.

Analogue An analogue scheme is created by using analogue colours. This means using colours which are next to each other on the colour wheel. When using an analogue scheme, one has to take into account that warm and cool colours should not be mixed with each other. The use of too many hues is also not advisable because of the danger of creating a too variegated presentation. In Figure 4.21 there is a warm analogue scheme created by only using red, orange and yellow.



Figure 4.20: Analogue

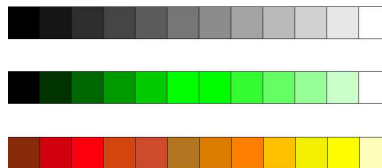


Figure 4.21: Examples of achromatic, monochromatic and analogue colour schemes.

Harmony through the equilibrium state of the eye. The three schemes described above are based on Tufte's minimal effective difference theory. The following harmony schemes are of a different type.

Most people probably know the illusions where we have to focus or stare for about 30 to 60 seconds at an image and then, when we look at a blank paper we can see the afterimage. If one tries to focus on the afterimage one can see it is the negative of the image one has been staring at. Even with coloured images this works and we see the complementary colours as a result. So if one focuses on a red spot, see Figure 4.22, and then looks at a

white paper one will see cyan, the complementary colour, for a short period of time⁸. This effect is due to the eye which seeks to restore its equilibrium. The effect of the afterimage that is created by the eye is called successive contrast.

If we look at Figure 4.23 we can see that the appearance of a colour is affected by what surrounds it. The image in Figure 4.23 only uses single colours red, yellow and blue. This effect we call simultaneous contrast.

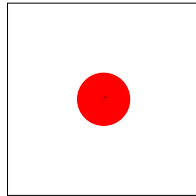


Figure 4.22: Successive contrast

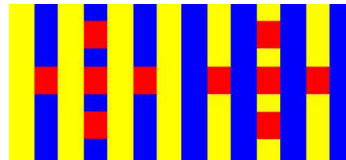


Figure 4.23: Simultaneous contrast

Both kinds of contrast indicate that the eye tends to create an equilibrium by using complementary colours. According to [21], Count Rumsford published in 1797 an article in the Nicholson's journal about creating colour harmony. He stated that colours are harmonious if they are white when they are mixed. It is obvious that he was talking about an additive colour model.

When we focus on a black spot on a white background we get a white spot as an after-image, the same for a white spot on a black background, this produces a black spot as an afterimage. This means the eye still seeks its equilibrium. The result of the research of physiologist Ewald Hering was that grey matches the equilibrium of the eye [21].

Colour harmony can thus be achieved by creating an equilibrium state for the eye, so the addition of the colours needs to be greyish. There are however several ways to create a harmonic colour representation. In the next paragraphs several different *Colour Harmony Schemes* will be described.

Complementary This scheme is based on the use of two different hues which are opposite to each other on the colour wheel and thus produce a high contrast, for example yellow versus blue, see Figure 4.24. When looking at the 3D HSL model we can pick a point in the model and create its complementary harmonious colour just by mirroring this point in the centre point of the 3D model. So the hue value of the colours differ by $180^\circ \pmod{360^\circ}$ and the lightness is mirrored on the XY plane. In this colour scheme it is advisable to use cool colours as foreground and warm colours as background. When using colours with a low saturation level, the colours will be very near the centre point of our HSL-space, these colours will look very similar and the combination of these colours is not advisable. A more flexible complementary scheme is the split complementary scheme, see Figure 4.25. Instead of using a single hue we use a hue range for both colours where the centre points of these ranges are complementary. Here one should avoid the use of highly unsaturated colours as well.

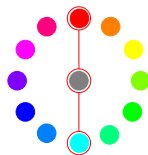


Figure 4.24: Complementary

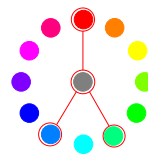


Figure 4.25: Split-Complementary

⁸The best way is to place this figure on a well lit stable area and to focus for 30 seconds on the little mark in the centre of the red circle. When looking at a blank paper, one should be able to see the cyan afterimage, this can be intensified by blinking the eyes.

Triad A triadic colour harmony scheme consists of 3 different hues, see Figure 4.26. If we take the colour wheel and place an equilateral triangle we have a harmonious triad. An example is the combination of the three primary colours, red, green and blue. Other triadic schemes are created by turning the equilateral triangle on the colour wheel. Each of the three hues selected to create the harmony scheme can be used as the centre of a range of hues.

To give this scheme some extra flexibility we can also use an isosceles triangle instead of an equilateral triangle. We then have to take into account that the angle of the top of this isosceles triangle must be in the range from 0° till 90° , because of the equilibrium constraint. With this triadic colour scheme we need not use only the colour circle only, but can use the whole dodecahedron to create a triadic colour harmony scheme. We only have to keep in mind that the point of intersection of the bisectors of the sides from the isosceles or equilateral triangle lies at the centre of the dodecahedron.

When using black, we ought to use two complementary colours very near to white, but not completely white, the same applies when using white, one should use two complementary colours very near to black.

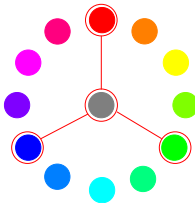


Figure 4.26: Triadic

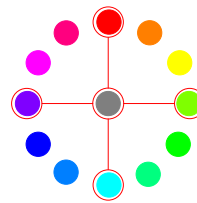


Figure 4.27: Tetradic

Tetrad The tetrad colour harmony scheme is based on the use of 4 hues, where we have two pairs of complementary hues, see Figure 4.27. If we connect these 4 hues it is permissible to have a square, rectangle or a trapezoid for still being a colour harmony scheme. Here we also do not have to stick with the single hues but we can create a symmetrical range of hues around the chosen hue. The 4 hues can also be rotated in the dodecahedron keeping in mind the symmetry of the hues relative to the centre point of the dodecahedron.

Pentad A pentad harmony scheme can be created by using an equatorial triadic harmony scheme and by adding white and black. The created colours are still in balance because we added black and white.

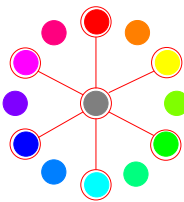


Figure 4.28: Hexadic

Hexad The same as being said for the pentad harmony scheme can be applied to the hexad colour harmony scheme, see Figure 4.28, if we take an equatorial tetrad harmony scheme and add white and black we retain a harmonious colour scheme. Another way of creating a hexad scheme is by drawing a hexagon in the colour wheel, by turning this hexagon in a dodecahedron we can get a wide variety of harmonious hexad colours.

Harmonic colour schemes in practice Colour schemes are useful for the interpretation and generation of images or documents. A well known example of the use of a colour harmony scheme is the work by Piet Mondriaan. In Figure 4.29 from [10] (page 127) we can see a pentad harmonic colour scheme. Many paintings can be captured in one of the above described harmony schemes. Van Gogh's painting in 4.30⁹ can be described as using a tetrad scheme where he is using, cyan, red, blue and yellow. The same painting can also be described in a split complementary scheme, where yellow and blue are the two *main* colours.

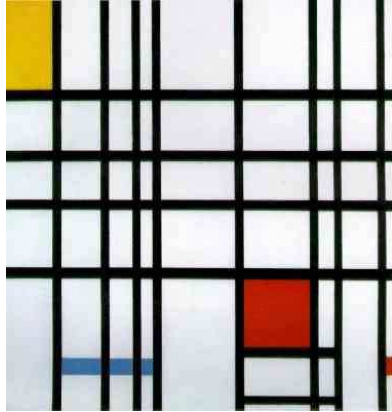


Figure 4.29: Composition with Yellow, Blue and Red, Piet Mondriaan, 1939-42



Figure 4.30: Harvest in Provence of Wheat Field with Sheaves, Vincent van Gogh, 1888

If we want to use harmonic colour schemes for making design decisions, we first have to know which scheme we are planning to be used. A guideline for choosing the scheme is the number of colours we want to use. Before doing this we first have to define some terms:

fixed colour A colour without any flexibility we will call a fixed colour, e.g. a corporate identity colour `hsl[250,255,128]`.

colour A colour is defined as a single hue, the different tones, tints and shades are all classes of this colour.

colour range A colour range is a range of different colours, this means a range of hues with their possible ranges of saturation and lightness.

relative colour Relative colours are a subset of the class of colour ranges. This subset can be given a single name which applies to its whole colour range, e.g. blue, orange, lilac.

If we, for instance, want to use 2 colours we can choose an analogue or a complementary scheme, for three colours we can use the triad scheme, but the analogue as well. This might sound arbitrary but every design decision process can usually be based on already established colours, such as a preferred colour, coming from the user model or the domain model (for example a corporate identity colour). When using such inputs the colour harmony scheme is almost provided. If the user, for instance, wants red and the corporate identity colour is a fixed blue and we need 4 colours, we can only use a tetrad harmony scheme, with blue, yellow, red and cyan. If we, however, needed three colours, then a triadic colour harmony scheme would have been a solution, but a very wide analogue scheme would have been a possible solution as well.

⁹from: <http://www.art.com/asp/sp.asp?PD=10056399&RFID=864458>.

We now have set a basis for graphic design using colours. Before we can continue with creating explicit design rules, we have to take an important factor into account, namely typography.

3. TYPOGRAPHY

In the previous sections we started with the basic colour theory and ended with harmonic colour schemes. Unfortunately, we can't apply all these schemes when we use text in a presentation. In this section we try to describe why certain rules don't apply.

The most important constraint of text on a certain medium is its legibility. If we place text somewhere, we want the viewer to be able to read it. One of the things we should not forget is the difference between text on paper and text on a screen. Text on paper is *solid*, whereas text on a display is based on the emittance of light waves. The theory stated below, however, applies on both mediums for displaying text.

3.1 Legibility

When dealing with text, one often thinks in terms such as font types, font sizes and font styles. Because of the different assumptions one makes when talking about legibility, we first clarify some terms:

- **typeface**
A typeface is a design for a set of fonts, and often comes as a family of typefaces, e.g. bold, italic and other variations.
- **typeface family**
A family of typefaces is thus a set of typeface variations.
- **font**
When giving a typeface a certain size we have obtained a font. A font therefore is a set of text characters in a specific style and size.

Summarizing the above definitions in an example, it can be stated that: Courier is a typeface family, Courier italic is a typeface and Courier italic 12pt is a font.

Returning to legibility we can state that overall legibility depends on five basic factors [23]:

1. **The typeface family**
Typeface families are often specifically created for on-screen or on-paper use. On-paper typeface families are mostly serif¹⁰, whereas on-screen typeface families are mostly sans-serif¹¹.
2. **The typeface**
When using italic typefaces for longer texts on screen or on paper, it can be very hard to read, when using bold typefaces, however, the legibility can improve. This depends mostly on the typeface family and size of the used typeface.
3. **Fonts**
The size of the typeface is a crucial factor for legibility. Because of the very low resolution of screens (around 72 dpi) compared to printers (up to 2400 dpi) very small typefaces can be very coarse on-screen while perfectly readable on paper.
4. **Font composition**
Elements of font composition are for example spacing (between characters and between lines) and the length of a line. Spacing shouldn't be too small or too large and the number of words per line, on A4 sized paper, should be 10-12 [37]. Factors, such as output medium type or font size influence this number of words per line.

¹⁰A serif is the little extra stroke found at the end of main vertical and horizontal strokes of some typeface families.

¹¹sans: without (French)

5. Font colour

The colour of a font is perhaps one of the most important factors of legibility. We ought not to forget that we are not only talking about the colour of the font itself but about the background colour of the font as well. When we are talking about background and foreground combinations, one have to keep colour harmony in mind.

All these factors are somehow dependent on each other. The typeface Geneva normal is, for instance, on-screen perfectly legible from 9 pts whereas the typeface Frutiger normal is legible from 12 pts. Another example is when the size of a font decreases. Keeping the text legible we have to increase the saturation of the colour of the font [11]. There is thus a wide variety of possible combinations to create a legible text. Time, however, sets limits on the scope of this research. This is the reason that the emphasis will be on the font colour for creating legible texts, see section 3.2. We, therefore, have to fix the other basic factors for creating legible texts.

As mentioned, sans-serif typeface families are often used on-screen, because the serifs tend to disappear on the computer screen. Serifs, however, aid the reader by giving more visual information about the shape of the letter and thus for legibility. Georgia, a serif typeface family, was specially designed for the screen. The typeface family looks good on paper and on screen, this is the reason that in our further research this typeface family is used. Figure 4.31 shows different fonts of the typeface Georgia regular.



abcdefghijklmnopqrstuvwxyz
ABCDEFGHIJKLMNOPQRSTUVWXYZ
1234567890©@

abcdefghijklmnopqrstuvwxyz
ABCDEFGHIJKLMNOPQRSTUVWXYZ
1234567890©@

abcdefghijklmnopqrstuvwxyz
ABCDEFGHIJKLMNOPQRSTUVWXYZ
1234567890©@

abcdefghijklmnopqrstuvwxyz
ABCDEFGHIJKLMNOPQRSTUVWXYZ
1234567890©@

Figure 4.31: The Georgia regular typeface in different fonts

3.2 Font Colour

Most people have a good intuition about what (not) to use for text and background colours. Here we try to capture this intuition and try to explain why we can or cannot use certain combinations. In the next section some general text colour rules will be derived from this knowledge, based partially on [11].

When we use colours for text and background it is easy to make bad legible combinations, here are some in the additive colour system:

- Yellow text on a white background.
- Blue text on a black background.
- Red text on a cyan background.
- Blue text on a red background

These are just a few of the infinite possibilities of creating a bad legible text! If we recall the colour harmony schemes, we can see that these combinations all fit in a certain scheme. Yellow on white and blue on black both fit in a monochromatic harmonic colour scheme,

whereas red on cyan fits in the complementary harmonic scheme. To fit blue text on a red background we can use a triad scheme if we are allowed to use a third colour: green. But why do the above examples create illegible combinations?

- Yellow text on a white background.
Both yellow and white are bright colours, if we recall the YIQ model we can calculate the Y component for both colours. If we assume we use fully saturated yellow we've got a relative Y component of 0.886, where the relative Y component of white is 1.000. The problem here is that the distance $|Y_1 - Y_0|$ is too small.
- Blue text on a black background
The same here as with yellow on white, except we have two dark colours, black has a Y component of 0.000 and the relative Y component of fully saturated blue is 0.114. Here we can also conclude that $|Y_1 - Y_0|$ is too small.
- Red text on a cyan background.
It was already mentioned that this combination could fit in the complementary colour harmony scheme. Using this scheme, however, for text and background colour is not advisable. The result of using complementary colours is a flickering effect¹², which makes the text harder and more tiring to read. On paper this should also be avoided (e.g. green on red). This effect can be weakened in various ways. One is for instance by darkening the colour by adding black, another way is by desaturating the colours.
- Blue text on a red background
Blue is obviously a cool colour, whereas red is warm. As already described in section 2.4, cool colours tend to recede from the viewer and warm colours tend to move toward the viewer. This makes warm colours more suitable for text than cool colours, see also [11].

When using bright colours with dark colours we have to consider the effect called *type glow*. This is the effect which only appears in the additive colour system where the edges of fonts appear thinner than they really are. This is because white or very bright areas are much brighter on screen than on paper. An example is the use of a white background and the use of black fonts, we have to avoid the type glow effect.

A study from 1990, however, which included visual search and reading, where accuracy and timeliness of response were measured, showed an increase of performance, in the range of 2.0% to a high of 31.6%, was obtained when using negative contrast (dark colours on a light background) instead of positive contrast (light colours on a dark background) [4]. Earlier studies also showed negative contrast gave better performance than positive contrast [46].

With the above knowledge we can create some general rules for creating legible text.

3.3 General typography colour rules

In this section we try to define some general colour rules.

- When using text we have to avoid colours with a low relative difference in luminance. Too much difference can also have a negative effect on legibility: $x_1 \leq |Y_1 - Y_0| \leq x_2$. The actual values of x_1 and x_2 have to be determined by experimentation.
- When picking a background colour, one has to avoid bright colours (afterglow).
- The use of a complementary harmony scheme, with saturated colours should also be avoided.
- We prefer negative contrast (dark foreground, light background) rather than positive contrast [46, 4].

¹²Because the eye is trying to focus on 2 different wavelengths

- Finally, when using a warm/cool colour contrast, use warm colours for the foreground and cool colours for the background.

4. CONCLUSION

Colour graphic design is a very broad topic, especially when considering the physical, biological and cognitive aspects. When reading elsewhere on the same subject one should keep the above theory in mind, because when doing this research many errors were found on the web and in books. Even on page 18 in [21], considered as a *bible* on colour theory, a fundamental error was found. It was stated that the complementary colours of green and yellow in the prismatic spectrum were red and violet respectively. This does apply to a subtractive colour model, but not to the prismatic spectrum which is an additive colour model (recall the RGB and CMYK models).

When dealing with colours in graphic design it is inevitable to treat the use of text in a presentation on screen differently. Colour harmony schemes can't guarantee that text is legible, and, whenever we use a colour harmony scheme, we have to distinguish between the additive and subtractive colour schemes.

The theory described in this chapter provides enough knowledge about graphic design to create a system which can perform colour design decisions automatically, given some constraints as input. We got colour harmony schemes to select a harmonious colour combination from and we got some global constraints for colours when using text in a presentation. In the next section we describe the architecture of the Design Module that takes the established rules of this section into account.

Chapter 5

The automated selection of colours

1. INTRODUCTION

The previous chapters provided us with background information required for the automatic generation of colour design. Chapter 2 gave us information about the underlying theory with respect to hypermedia documents. In chapter 3 the automatic presentation generator Cuypers, its overall architecture and the 5 associated presentation generation modules were described. Chapter 4 explained graphic design theory and legibility factors, focussing on colour design aspects. The following chapters will combine these established theories to provide an architecture for the Design Module. Furthermore they describe the actual realisation and implementation. This chapter proposes an approach towards automatic colour design by using an example scenario to clarify the architecture of the Design Module.

Research on automated colour design, in particular on colour group selection for computer interfaces [27] has already been carried out. This research, however, used a given diagram on which various colour schemes were tested, in contrast to the dynamic environment of our prototype.

2. EXAMPLE SCENARIO

Before realising the Design Module we need to know what the Design Module actually does. This and the following section will provide the base on which the realisation of our approach on automatic colour design can be built¹.

The process of designing visualisations of information, be it for cultural or technical purposes, is complex and resource demanding, where the end result might establish various presentational forms, such as textbook illustrations, advertising images, or a fine art multimedia presentation.

Consider the introduction screen for a presentation on the art movement “De Stijl”, as portrayed in Figure 5.1. This page emerged from the wish of a visitor of the web environment of a museum to learn more about the “De Stijl” movement after she was intrigued by a Mondrian image she found while browsing the collection.

As well as decisions about the overall logic of the presentation, for example that further investigation of the subject or related topics should be possible, a considerable number of decisions need to be made with respect to layout, typography and colour design of this particular page to ensure its functionality. The prerequisite of legibility of text, for example, might result in a provision of black text on a white background. Such a decision then

¹This section and the next section are adopted from [33].



Figure 5.1: Introduction page for a presentation on the art movement “De Stijl”.

requires that the different functionality of textual links can be easily distinguished, for example by assigning a particular colour scheme to them. On an aesthetic level the page establishes a “De Stijl” feel by advocating pure abstraction and simplicity: form is reduced to the rectangle, typography for headers uses the “De Stijl” font set, and colour uses the primary colours red, blue and yellow, along with black and white.

Having introduced the main components within the Cuypers engine in chapter 3, we are now in a position to explain our approach that addresses the initial requirements for the balance between functional and stylistic aspects in dynamic multimedia presentations, focusing on the automatic design of the colour scheme for a presentation. We base the discussion in the context of the generation of a presentation about the art movement “De Stijl”.

The user was browsing the collection and came across a Mondrian image resulting in a request for more information about the “De Stijl” movement. This request is interpreted, based on the user model, as the wish for a presentation that allows browsing through an evolving presentation. By evolving we refer to the concept of progression of detail that facilitates navigation based on a given weighted set of descriptors representing a story context on a micro level (next step in content exploration) as well as on a macro level (larger contextual units clustering content, such as classes of artefacts within an art movement), as described in [14].

In the following section the colour design process will be described in detail.

3. AN APPROACH TOWARDS AUTOMATIC COLOUR DESIGN

Our discussion starts at the point where the presentation engine has established the constraint-set on which the colour design process is based, as shown in the top line of Figure 5.2.

3.1 Constraints driving the colour design process

In this section we give details about the constraint-set provided by the presentation engine. The required steps up to this point, such as the design of the discourse structure, the semi-automated generation of metadata and the content retrieval process [34, 32], as well as the automatic design of the spatial layout for retrieved material [15] are described elsewhere.

A typical request rule looks like this:

With this rule, the generation engine requests from the design engine a solution for a

```

design_module(  Design_task,
               Communication_preference,
               Constraint_list,
               Result_list).

```

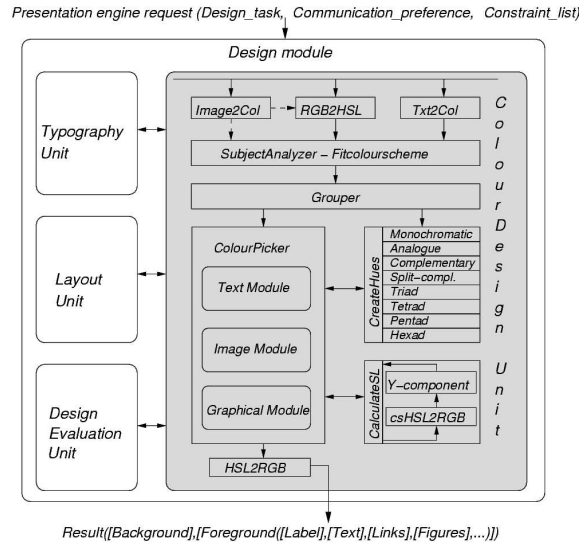


Figure 5.2: Detailed view on the colour component of the Design Module

particular task based on the provided constraints. For the current example, the design-task is “colour_setting”.

The request further determines a communication preference for the presentation that was already established in earlier design stages. The identified preference guides the graphic design process by suggesting which general design orientation the process should follow. In our environment we utilize the following three: **function**, **form**, **neutral**.

A preference for **function** determines that constraints based on importance weights are driving the design process. For the colour design process this might result in the dominant application of rules that compare colours based on contrast, as this effects the construction of legible text. On the other hand, favouring **form** will mainly employ rules that effect the appearance of colours, such as for achieving pleasing appearances based on colour harmony schemes as described in section 2.5. **Neutral** means that either way is acceptable. In our example, where the presentation is aimed at an educational presentation, a functional colour design process is required.

The `Constraint_list` contains the relevant restrictions for the current design task, namely “colour setting”, established during previous generation stages performed by other generation modules. Such constraints supply a means of controlling the number of colours to be used and, even more important, the automatic allocation of colours to the various elements provided by the spatial layout of a page, as presented in Figure 5.3.

The structure of the `Constraint_list` introduces particular constraints related to the user, the domain and the layout, where each part represents in detail:

User (provided by the user module)

Physical_abilities that describe possible audio-visual deficiencies, such as colour blindness. Colour blindness makes it impossible to use red and green as structural distinguishes on the same page, as the user cannot distinguish between both colours. In our system colour blindness is characterised as `green_red`.

Domain (provided by the domain ontology and user module)

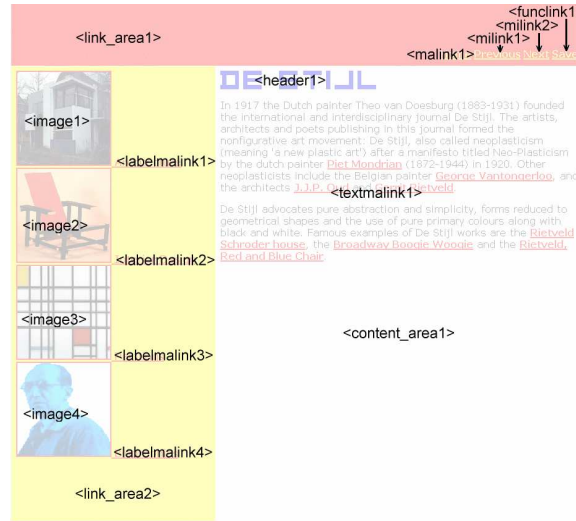


Figure 5.3: Graphical representation of the spatial layout structure before the colour design process.

```
constraint_list(
[green_red, [red, blue, yellow],
[[area: [content_area1: 1.0], elements: [textmalink1: 1.0, header1: 1.0]],
[area: [link_area2: 0.9], elements: [labelmalink1: 0.8, labelmalink2: 0.8,
labelmalink3: 0.8, labelmalink4: 0.8, image1: 0.6, image2: 0.6, image3: 0.6,
image4: 0.6]],
[area: [link_area1: 0.4], elements: [malink1: 0.3, milink1: 0.8,
milink2: 0.8, funclink1: 0.8]]]]).
```

Figure 5.4: Example of an instantiated Constraint_list

Domain.colour is a list of colours that are related to the domain the content is about or to particular colours required by the content environment the user is using, for example corporate identity colours. In our example the system might pick the primary colours used in the domain ontology for describing the “De Stijl” colour scheme [red, blue, yellow].

Layout (provided by the Layout unit of the Design Module)

Presentation.elements, which is a structured list of all areas available and the various information units combined in them. The list reflects on the one hand the spatial layout through ordering the various presentation areas according to their size (the largest area, which is with respect to colour the most dominant, comes first). On the other hand it also provides information about the importance of each information item according to its role within the discourse structure, qualified by an associated importance weight. Information items in this context can be: text, text containing links (textmalink), header, label, label as link (labelmalink), links for navigating on a micro level, such as moving from a definition to an example (milink), links for navigating on a macro-level, such as jumping from the architecture section to the painting section (malink), functional links, such as search, print, or save (funclink), images, etc. An instantiated Constraint_list is shown in Figure 5.4.

The educational basis of the communication goal of our example presentation is represented in the high importance value of the textmalink element within content_area1. Moreover, as the navigation style of the presentation is set to be “evolving”, the importance for macro- and micro-navigation is reflected in the high importance value of the various navigation links.

The `Result_list` determines the output of the Colour Design Unit, namely the description of areas and included elements with the corresponding colour allocations. The presentation engine then processes the outcome to generate the final presentation of that page.

Based on the given description of the input parameters of the Colour Design Unit we can now demonstrate how the module solves the established constraints and creates the colour scheme for the presentation.

3.2 The generation of the overall colour design

The Design Module, as described in Figure 5.2 on page 41, covers various design aspects, such as typography, layout or colour-design. In addition to those rather specialised units within the Design Module, the Design Evaluation unit provides high-level rules, which are used to guide the design process or to evaluate intermediate results as a form of sanity check. Such rules might state, for example, with respect to the colour design process that:

- A. If `Communication_preference`
 = `function` => work on area elements
 = `form` => work on area
 = `neutral` => work on mix
- B. If there are multiple background areas, begin assigning the one with the largest size first and continue in descending order.
- C. The number of colours used ≤ 6 .

Rule A determines whether the emphasis of the colour design is oriented towards the functionality of the information elements (foreground and structure) or rather on a look and feel, emotionally oriented presentation (emphasis on background and contrast).

Rule B considers the overall importance the amount of space within colour design. Larger areas are dominant with respect to perception and thus more influential for the design process than smaller areas. Note, there are other rules related to space that consider the shape of an area and the depth of colour in colour combinations (foreground background relationship) (see [21], pages 120-124, 144-149).

Rule C provides the maximum number of colours to be used in a presentation. Tests described in the literature showed that too many and different colours on a screen attract the user's attention to the colours themselves and distract from the content [11, 12]. We experimented with a range from 4 up to 8 colours and found that the system could work most effectively with 6. More user testing is required, though, to finalise this rule.

These rules are required to organise the different steps within the colour design process, as described in Figure 5.2 on page 41, which can be separated into:

- identification of available colours;
- detection of the colour scheme to be used;
- instantiation of the process order for the colour assignment;
- colour assignment.

These steps are described in more detail in the following sections.

Identification of available colours The first step of the Colour Design Unit is to identify the potential number of colours to be used. For that the SubjectAnalyzer (see Figure 5.2) evaluates the `constraint_list` on areas and the information units contained in them. For the example provided in Figure 5.4 the SubjectAnalyzer comes up with 8 colours, namely three colours for the areas, three colours for the different sorts of links (mi-, ma- and funclinks), one colour for the text part of the textmalink field and one colour for the header.

High-level rule C states, however, that the overall number of colours should not exceed a total of 6. Thus the SubjectAnalyzer re-evaluates the constraint-set by trying to group units. Merging the three types of links into one presentational group provides the required result and thus the number of colours to be used is established as 6.

Additionally, the SubjectAnalyzer identifies already selected colours, provided by the `Domain_colour` element of the `Constraint_list` [red, blue and yellow]. Note, in our implementation the colours have already been transformed to actual HSL values using the “Txt2Col” transformation module (see top of Figure 5.2 on page 41), which transforms textual colour representations into the numeric expression in HSL format of the corresponding fundamental colour. For example, red is transformed into `hsl[0,255,128]`.

At this stage it is known that no more than 6 colours need to be associated and that three of them have been determined.

Detection of the colour scheme The next step is to identify the colour scheme used to assign colours. In our system four different schemes are implemented, namely achromatic, monochromatic, analogue and complementary (including split complementary), where each scheme represents a particular rule set for colour use.

Applying our implemented schemes to the example scenario, the SubjectAnalyzer now tries to establish the scheme used to pick and allocate colours for the presentation.

The selection process applies the approach of growing complexity to support the time constraints of the overall presentation generation. Thus, the algorithm tries to apply the achromatic scheme first, as it provides the smallest rule set. However, it fails because this scheme does not support colour. The monochromatic scheme is rejected because it allows only a single hue, whereas the domain colours already provide three different hues. The analogue scheme cannot be selected for several reasons, such as

- the covered distance between the two outermost domain colours (red and blue) is too large (the angle between red and blue over yellow is larger than 120° in the HSL space),
- cool and warm colours would be mixed (blue with red or yellow),
- the distance between red and blue over yellow potentially allows the use of green as additional colour, which is not possible, as the user has extreme deficiencies in distinguishing between red and green.

Finally, among the available complementary schemes a split complementary scheme is chosen (red and yellow grouped on one side, blue on the other within the three dimensional colour space), as it facilitates the use of the established colours.

Instantiation of the process order for the colour assignment The next step in the colour design process is to establish the order in which the different areas and their elements should be coloured. The order is of importance, as assignments for the most relevant area should be based on the largest colour set (see rule A in Section 3.2). Established assignments will then act as further constraints in the ongoing colour picking process. The Grouper, as presented in Figure 5.2, performs this task.

There is one essential assumption on which the performance of the Grouper is based, namely that areas usually provide the background within a presentation, whereas the associated elements supply everything that is layered on top of it (foreground). The Grouper can now either pay special attention to the size of the areas (form), to the importance weights for the associated information units (function), or a mix of both (neutral).

Emphasising does, however, not mean neglecting other relevant elements. For example, as the physical size of an area specifies its importance for the feel of a presentation (the larger an area the more dominant its colour), it has to be considered even though the main orientation of the presentation is functional. Thus, in our experimental system we work

with three different ways of calculating the importance of an area, each taking both the size of the area and the contained information elements into account.

$$I_{AT_{function}} = \frac{(I_A \times P) + \left(\frac{\sum_{i=1}^n I_{E_i}}{n} \right) \times 2}{2}$$

$$I_{AT_{form}} = \frac{(I_A \times P) + \frac{\sum_{i=1}^n I_{E_i}}{n}}{2}, \text{ with } I_A = 1$$

$$I_{AT_{neutral}} = \frac{(I_A \times P) + \frac{\sum_{i=1}^n I_{E_i}}{n}}{2}$$

I_{AT} represents the overall importance of the area, I_A stands for the importance weight of the area with respect to content, P represents the size of the area provided by the reverse position value in the `constraint_list`, and I_E stands for the weight of an individual presentational unit.

We are aware that the current formulae do not fully represent the complex relationship between the foreground elements themselves, or all aspects of the relationship between foreground and background, especially when it comes to various layers of foreground elements. However, trials with changing numbers of areas (max 5), associated elements (max 20) and various levels of importance (in the range between 0.1 and 1.0) showed that the formulae establish a sensible hierarchy of areas, where areas are organized in decreasing order of importance value. Note, it does not necessarily mean that the order is different from the one already provided by the `constraint_list`, which is only based on the size of areas.

As the `communication_preference` of the ongoing example is set to `function`, the Grouper uses the formula $I_{AT_{function}}$, resulting in the following order of areas: `Content_area1`, `Link_area1`, `Link_area2`.

Once the order of areas is established the Colour Design Unit starts with the assignment of colours to the most important area, in this case `Content_area1`.

Colour assignment The relevant unit for this task is the ColourPicker as shown in Figure 5.2.

Colour assignment colour_area1

The ColourPicker first separates background from foreground elements.

As functionality is more important, the ColourPicker tries to identify the colour for the foreground elements in `content_area1`, i.e. the `textmalink1` and the `header1` element. As both elements are of type text, the TextModule of the ColourPicker attempts to establish the element colours using those colours that are already available: namely red, blue and yellow.

It is important to notice that the TextModule establishes simultaneously the background colour of this area too, as both colours depend on each other. The emphasis in the colour design process lies, nevertheless, on the foreground.

The TextModule uses the domain specific colours in their HSL notation and tries all possible colour combinations with respect to all foreground and background groupings, where each combination is assigned with a legibility value (LV) based on

- the relative brightness factor ($Y_{colour1} - Y_{colour2}$)
- the contrast between both colours based on their classification as light and dark colours (light colours should be used as background and dark colours for the foreground on the screen [4])

- the contrast between both colours based on their classification as cold and warm colours (cool colours should be used as background and warm colours for the foreground)
- the complementary contrast between both colours.

The calculated LV lies between 0.0 and 1.0 and the higher the value, the better the combination.

Note, the above approach does not use colour ranges as input. The reason for not allowing that is that usually the provided colours have a particular value that must not be altered, such as the domain colours for a particular style or art movement or corporate identity colours. If the colour Design Module chose freely, ranges would be useful but we do not explore this direction.

For our example the combination of a yellow background and a red foreground provides the highest LV (0.633). However, the LV does not reach the threshold of 0.8 that is required by the DesignEvaluation unit to accept a colour combination in the function mode.

As the result the TextModule falls back to the default colour for text, namely black, which now also sets the foreground colour for that area. The next colour to be assigned is the background colour. Within the split complementary scheme, the complementary colour for black in the HSL space is white.

Once both colours are picked the DesignEvaluation unit performs a final test on lightness and contrast required for achieving “legibility”. In fact, the extreme contrast between black and white should be avoided, as it can create extra shimmering effects that can strain the eye. Thus, the contrast-smoothing rule is applied, which adds 10% of each colour to the other, resulting in two grey values that still give the impression of being essentially black and white. However, both originally chosen colours will be added to the colour selection (i.e. red, blue, yellow, white and black = 5).

As the text body also contains links the TextModule next assigns the colour for the link. The aim is to distinguish link and text from each other. The goal is to find among the depicted colours the one that provides the largest contrast to both the foreground and the background. In our example the best distance within the HSL colour space to black and white is provided by red. Thus, red will be assigned to the link. Note, the link colour comes with two values. One represents a non-visited link and the other a link that was already visited. The TextModule applies one of the split-item strategies for assigning two colours to a link. This rule diminishes the saturation of the established colour by 50%, which provides a sufficient difference to present the two states of a link.

Finally the Colour Design Unit has to assign a colour value to the “header”. The reasoning is similar to the one described for the process of colouring links. The result is that the header is assigned to blue because it provides a larger distance to white (background) than yellow.

At the end of the colour design process for the content area we have the following situation (text in brackets is added for better readability of this document):

Result_list:			
Background:	content_area1:	hsl[0,0,230]	(white-grey)
Foreground:	content_area1:		
	textmalink1:	hsl[0,0,25]	(black-grey)
	link:	hsl[0,255,128]	(red)
		hsl[0,128,128]	(0.5 red)
	header:	hsl[240,255,128]	(blue)

Colour assignment link_area2

The next area to be coloured is Link_area2, as described in Figure 5.3 on page 42. As the

process will use already established colour allocations, the ColourPicker takes the colour for link as given. The ImageModule considers images as being of colour “various”. This is a default within our current implementation, as the solution of matching the background with visual material such as images or a video area, is not fully implemented yet. Details about the approaches taken are described in section 5.

Thus, the final decision to be made is concerned with the allocation of the background colour of this area. As the colour needs to be distinguished from the neighbouring area (white) but has to allow the legibility of the links (red) the TextModule decides on yellow, mainly because

- the combination of blue as background and red as foreground result in a too small LV (0.178) to be acceptable and
- the contrast difference between yellow and white is big enough to allow a differentiation between areas ($Y_{white} - Y_{yellow} > 25$).

At the end of this part of the colour design process the system has made the following allocations:

Result_list:			
Background:	content_area1:	hsl[0,0,230]	(white-grey)
	link_area2:	hsl[60,255,128]	(yellow)
Foreground:	content_area1:		
	textmalink1:	hsl[0,0,25]	(black-grey)
	link:	hsl[0,255,128]	(red)
		hsl[0,128,128]	(0.5 red)
	header:	hsl[240,255,128]	(blue)
	link_area2:		
	link:	hsl[0,255,128]	(red)
		hsl[0,128,128]	(0.5 red)

Colour assignment link_area1

The final area to be designed is Link_area1, as described in Figure 5.3 on page 42. Equivalent to the steps described in section 3.2 the system considers the colours for links set to red.

With the three unassigned colours, namely red, blue, and black, the TextModule is not in the position to resolve the general contrast constraints for background colouring applicable in the split complementary colour scheme. Hence, the Colour Design Unit backtracks and tries to apply another colour scheme to solve this problem. The choice is the analogue scheme, as it mainly addresses the combination of colours that are next to each other within the HSL colour space.

The first approach to assign blue to the background fails because the general rule applies that two dark colours (here blue and red) should not make up back- and foreground. The same applies to black.

The next option for the system is to find an area where

- the information unit link is used and
- a background colour is allocated that is within the neighbouring range of one of the colours within the analogue scheme.

As a result the system retrieves the allocations for the link_area2. However, the combination yellow (background) and red (foreground) cannot be used as no contrast between the two areas can be achieved (in fact the contrast is 0).

As the importance of `link_area1` is ≤ 0.5 (see Figure 5.4) and both red and yellow are in the domain set, the `ColourPicker` retracts to a shortcut rule that allows the reverse application of colours under these circumstances, resulting in an allocation for red to the background and yellow to the foreground. As the importance of this area is ≤ 0.5 , the `ColourPicker` rules out further functional evaluation. This is important, as the `ColourDesign` unit does not try to fill the still open 6th colour-slot.

The end result of the colour design provides the allocations as can be seen on the next page.

Result_list:			
Background:	content_area1:	hsl[0,0,230]	(white-grey)
	link_area2:	hsl[60,255,128]	(yellow)
	link_area1:	hsl[0,255,128]	(red)
Foreground:	content_area1:		
	textmalink1:	hsl[0,0,25]	(black-grey)
	link:	hsl[0,255,128]	(red)
		hsl[0,128,128]	(0.5 red)
	header:	hsl[240,255,128]	(blue)
	link_area2:		
	link:	hsl[0,255,128]	(red)
		hsl[0,128,128]	(0.5 red)
	link_area1:		
	link:	hsl[60,255,128]	(yellow)
		hsl[60,128,128]	(0.5 yellow)

The result list is returned as an answer to the request that was described earlier at the end of section 3.1. The transformation steps that generate the final presentation, e.g. generating a SMIL presentation, are described elsewhere [55]. Note, the established colour allocations will be used during the ongoing presentation as input for further alterations, as required.

4. CONCLUSION

The approach discussed in this chapter can be used as a starting point of realising the Design Module. The architecture is sufficient to let the system make the right choices and suggestions.

Any system goes through the phases of a life cycle one or more times during its lifetime [47]. The life cycle of a system consists of different stages, namely planning, design, realisation, implementation, exploitation and evaluation. This chapter covered the aspects just before realisation, which will be described in the next chapter. The example scenario where the user requested information about the “De Stijl” movement, can be used to evaluate the system’s decision process and its performance, as being shown in the next chapter too.

Chapter 6

Realisation and Implementation

1. INTRODUCTION

In the system life cycle [47] the next two steps after system design are realisation and implementation. Realisation means building the system according to its functional design, which is implicitly depicted in the previous chapter. Realisation includes testing the system to see if it meets its requirements. The next step is the implementation phase, which means preparing the system to take into operation, thus integrating the system into the environment in which it will work, entering necessary data and adjusting parameters. The first part of this chapter describes the realisation and the second part describes the implementation of the system into a web interface and into the Cuypers system.

2. REALISATION

The starting point of realizing the Design Module was the example scenario and the proposed architecture, as described in chapter 5. The actual realisation is, however, a generalisation of the scenario, as the system should be able to generally determine colour schemes. This section describes the actual components realised in the Design Module. We use, however, the scenario to explain the functionality of the components.

The Design Module was implemented in Prolog and Eclipse. The latter is used for constraint solving. The next section describes the entire process of assigning colours. The description of the Design Module follows the architecture, as depicted in Figure 5.2 on page 41.

2.1 Architecture and the components

In Figure 5.3 on page 42 we saw the different interface areas and elements the example scenario was built with. When realizing the Design Module it was convenient to build it standalone. Thus the *input* parameters were predefined and located in a prolog file called *pages.pl*.

The top level handler, *designer.pl*, initialises the program by calling *loader.pl*. This prolog file takes care of loading all different components of the Design Module. Then *designer.pl* retrieves the *constraint_list* as well as the orientation value for the form-function balance. In the current example the value is set to “function”, which indicates that the Design Module aims for a functional presentation. In the next step the *subjectanalyzer* analyses the retrieved information.

subjectanalyzer.pl

```

constraint_list(
[green_red,[red,blue,yellow],
[[area:[content_area/1:1.0],elements:[textmalink/1:1.0,header/1:1.0]],
[area:[link_area/2:0.9],elements:[labelmalink/1:0.8,labelmalink/2:0.8,
labelmalink/3:0.8,labelmalink/4:0.8,image/1:0.6,image/2:0.6,image/3:0.6,
image/4:0.6]],
[area:[link_area/1:0.4],elements:[malink/1:0.3,milink/1:0.8,
milink/2:0.8,funclink/1:0.8]]]]).

```

Figure 6.1: Instantiated Constraint_list with “De Stijl” data

This component first uses the `constraint_list` to see whether the user has any vision abnormalities, as described in section 2.2. It then translates the domain specific colours into HSL values by using a utility module *txt2col.pl*. It then analyses the layout structure with respect to the available number of areas and the type of presentation elements used. The final task of the `subjectanalyzer` is to initialise the search of colour schemes which fit the given domain colours.

The search of colour schemes which fit the given domain colours is done by the module *fitscheme.pl*.

fitscheme.pl

Fitscheme compares the given input colours and identifies the scheme the colours fit. In our case, there are three colours, namely red, yellow and blue. Contrary to the example scenario in the previous chapter, the program decides for either a tetrad scheme (with additionally cyan) or a hexad scheme (with additionally green, cyan, and magenta). These two schemes are the only ones that fit the three domain colours as can be seen in Figure 6.2.

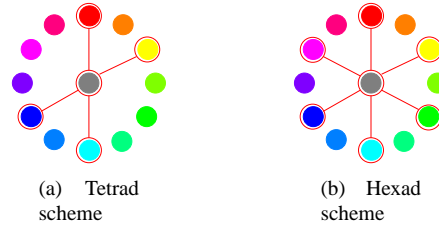


Figure 6.2: The two schemes which fit the three domain colours

Results from the `subjectanalyzer` are thus the translated domain output colours and the schemes these colours fit:

```

DomainColours = [[0,255,128],[60,255,128],[240,255,128]],
Schemes = [tetrads,hexads].

```

The *subjectanalyzer* returns the result list to the top level handler *designer.pl*. Here, the next task is to instantiate the process, which orders the identified areas to be coloured. The order in which the different components of the layout are labeled with colours is important for the final presentation. When areas and their elements are assigned with colours they influence the assignment of the following areas and their elements. The ordering process is performed by *processconstraintlist.pl*.

processconstraintlist.pl

The size of the different areas and the weight of the different elements in these areas determine their final order. The form function balance is also a factor

which influences the order of the layout data in the *Constraint_List*. The formulae are given in section 3.2, and are applied to the different areas of the example *Constraint_List*. As the overall design is “functional” the module uses the “function” formula resulting in weights of 2.5, 0.875 and 1.6 for *content_area/1*, *link_area/1*, and *link_area/2* respectively. The now ordered layout, which is the same as the original layout, is displayed below:

```
[[area:[content_area/1:1.0],elements:[textmalink/1:1.0,header/1:1.0]],
[area:[link_area/2:0.9],elements:[labelmalink/1:0.8,labelmalink/2:0.8,
labelmalink/3:0.8,labelmalink/4:0.8,image/1:0.6,image/2:0.6,
image/3:0.6,image/4:0.6]],
[area:[link_area/1:0.4],elements:[malink/1:0.3,milink/1:0.8,
milink/2:0.8,funcmlink/1:0.8]]].
```

The preparation phase has been finished by now and *designer.pl* can begin instantiating the colour picking routine. For that, it starts *colourpicker.pl*.

colourpicker.pl

This module, together with its submodules, is responsible for the actual assignment of colours for the background of an area and for the assignment of the colours for the different elements in the area. What it actually does is take each area from the *Layoutlist* and check for elementtypes in each area. The form function balance is responsible for the order in which elements will be searched for. In our case, where the balance is set to *function*, the picker starts with *content_area1* and searches for text elements, as text elements are important in conveying the communicational goal and thus by assigning them first supports the overall design which is “functional”. If the colourpicker found a text element it calls *assigntext.pl*, which is responsible for the text assignment. If the colourpicker did not find any text elements, it will continue with searching for link elements, which will be assigned by *assignlinks.pl* and finally the image elements, which will be assigned by *assignimages.pl*. If the form function balance is set on *form*, however, the picker searches first for images, links and then for text. If the form function balance is set to *neutral*, the order will be text, images and then links. After assigning all colours for one area the colourpicker continues with the next area, until all areas and elements are assigned in the order determined by the *processconstraintlist*.

The three submodules of the colourpicker are thus *assigntext*, *assignlinks* and *assignimages*. The next three sections will briefly describe these three modules. One has to be aware, that when there is no background colour set for an area, the first call to *assigntext*, *assignlinks* or *assignimages* will set the background colour for that area. The background colour is thus influenced by the order in which the three submodules are called.

assigntext.pl

Assigntext looks at the different combinations of colours provided, that is the domain and preferred colours, and chooses the best combination. This choice is based upon the ranking of all the text background combinations, which were determined in *textrank.pl*. If the best foreground-background combination exceeds a certain threshold, which is based upon the form function balance and the weight of the element, the foreground and background colour are set. This threshold is set in a prolog file called *config.pl* and this file contains several settings for “adjusting” the program. At the time of writing, these values are experimentally determined. If this threshold is not exceeded, *assigntext.pl* calls to *suggestor.pl*, which will provide *assigntext.pl* with a suggested colour. This module will continue suggesting colours which fit the schemes by means of backtracking, until the threshold is exceeded. If no colour is able to exceed

the threshold after trying all the possible schemes, the *suggestor* suggests the defaults, which are black on white. If the weight of an area is, however, too high, which is also set in *config.pl*, then the *suggestor* also suggests black on white, as defaults.

assignlinks.pl

This module provides the colourpicker with link colours. The first check is whether a link colour is already provided in that area. If so, the module will reuse the colour. If a background colour is provided, the module determines the biggest difference between the colours used in that particular area and the colours which are still left, that is, the colours left from the domain colours and preferred colours. If this list is empty, the module will call the *suggestor* which suggests a colour. If there is no background colour supplied the system checks whether in a previous area there is a link colour used and tries to reuse this colour with a background colour. If the weight of a particular area is below a certain threshold, the system tries to apply the *flipper* method. This means flipping the foreground and background colours of a previously defined area which included links.

assignimages.pl

Assignimages does not provide colours for the elements, but the colour for the background of an area. In the current implementation the colour of the images is hardcoded in the program. Future work involves extracting a generic colour from the images and based on this colour the module would attempt to provide a background colour.

There are two modules involved with the assignment of the different elements, namely *textrank* and *suggestor*. These will be described below.

textrank.pl

Ranking colour combinations on legibility is the task of this module. The input is a list of colours and this module ranks each combination of foreground and background colours. Thus, if we have n input colours this module evaluates $n \times (n - 1)$ combinations. At the time of writing the ranking takes all of the following aspects into account:

- differences between warm and cool colours, as warmer colours seem to move forward and cooler colours seem to recede, thus warm on cool combinations are ranked higher than cool on warm combinations.
- differences between light and dark colours, onscreen background colours needs to be darker than foreground colours (see section 3.3), thus dark on light are ranked higher than light on dark.
- complementary combinations, as complementary combinations with pure colours, meaning fully saturated and a lightness of about 128, make it hard to focus on the background or foreground colour. The result is a flickering effect which we have to avoid. If the combination tends to be a complementary one, the ranking will be set lower.
- colour-blindness checks.

Altogether means that the combination of e.g. yellow as a background colour and black as a foreground colour ranks higher than yellow as a foreground colour and black as the background colour. The values determining the weight of a decision result are stored in *config.pl* and thus can be adjusted.

suggestor.pl

The suggestor, as the names already implies, suggests a colour. The input is a list of colours and a list of schemes. The suggestor checks if the scheme fits the supplied colours and then suggests a colour which fits in the scheme. Currently the analogue, complementary, split complementary, triad and the tetrad schemes are implemented.

After processing all the steps with the given “De Stijl” input, we get the output as depicted in Figure 6.3. The colours in the output page are the same as in Figure 5.1 on page 40.

```
[[area : [content_area / 1 : 1.0 : [0, 0, 255]],
  elements : [header / 1 : 1.0 : [0, 0, 0],
    malink / 2 : 1.0 : [0, 255, 128], text / 1 : 1.0 : [0, 0, 0]]],
[area : [link_area / 2 : 0.9 : [60, 255, 128]],
  elements : [image / 4 : 0.6 : [20, 255, 128],
    image / 3 : 0.6 : [20, 255, 128], image / 2 : 0.6 : [20, 255, 128],
    image / 1 : 0.6 : [20, 255, 128], labelmalink / 4 : 0.8 : [0, 255, 128],
    labelmalink / 3 : 0.8 : [0, 255, 128], labelmalink / 2 : 0.8 : [0, 255, 128],
    labelmalink / 1 : 0.8 : [0, 255, 128]]],
[area : [link_area / 1 : 0.4 : [0, 255, 128]],
  elements : [funclink / 1 : 0.8 : [60, 255, 128],
    milink / 2 : 0.8 : [60, 255, 128], milink / 1 : 0.8 : [60, 255, 128],
    malink / 1 : 0.3 : [60, 255, 128]]]]
```

Figure 6.3: Output generated with “De Stijl” input

The suggested architecture, see Figure 5.2 on page 41, has been adjusted to the actual architecture and is portrayed in Figure 6.4. An extra input- and output line has been created. The input goes to the “pages” module and the XHTML+CSS output will be explained in more detail in section 3.

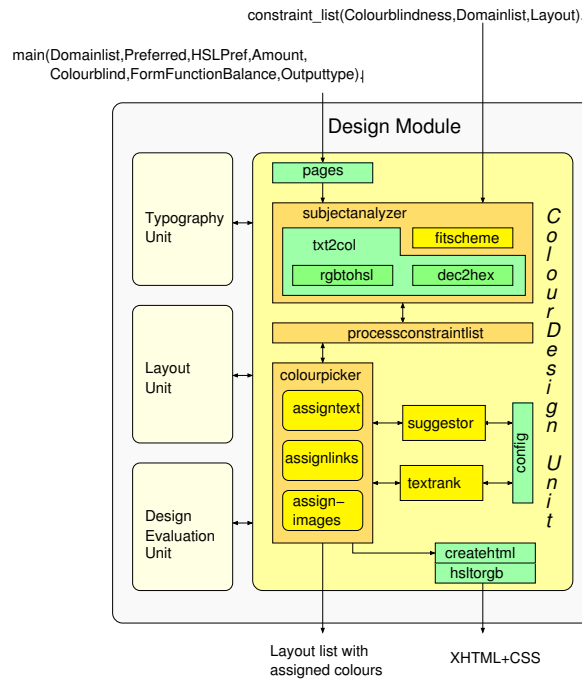


Figure 6.4: The actual architecture of the Design Module

In this figure we see a number of utility modules, namely *pages*, *hsltorgb*, *createhtml*

and *txt2col*, which uses *rgbtoshl* and *dec2hex*. These modules are not needed by the actual colour assignment of the various areas and their elements, but support the Design Module by doing pre- or post processing. A brief description of these utility modules will be given below, as well as a description of the *config* element which is used by the *suggestor* and *textrank*.

pages.pl

This prolog file provides information that otherwise would be supplied by the discourse-, platform-, user module or the domain ontology. As the following description of the architecture is based on the scenario example, see section 3 on page 40, the input file provides the user characteristics, the domain characteristics, the form function balance and the layout, together with the weight of the different areas and elements (see Figure 6.1 on page 50 for details).

hsltorgb.pl

Based on the algorithm in appendix 2 this module converts HSL encoded colours to RGB encoded colours. The HSL values for the hue are in the range from 0 to 359 and for the saturation and lightness from 0 to 255. The output RGB encoded values are in the range from 0 to 255. If the colours need to be used for the WWW, for example in a CSS, the values need to be converted to hexadecimal values. This is done by the *dec2hex* module.

createhtml.pl

This module is used when the webinterface is used and produces HTML output including CSS, which defines the colours that are chosen by the Design Module. More information about this web interface is supplied in section 3.1.

The *subjectanalyzer* uses *txt2col.pl* for translating colours described in natural language and in RGB values into HSL values.

txt2col.pl

When given a colour name, such as blue, yellow or red, this component translates these values to [240,255,128], [60,255,128] and [0,255,128] respectively. If a list of colours is provided, this module translates the entire list, and if HSL values are given, the module returns the value. The colours now known by this module are the 12 colours from the colour circle, where each colour represents 30° of the entire circle. Colours which aren't named are described by combining the two colours just before and after this colour (eg. [330,255,128] is called magentared). Black and white are also described with values [0,255,0] and [0,255,255] respectively. The names and HSL colour values can be easily extended. Whenever there is an RGB encoded colour given, characterized by a string preceded by a “#” sign and with a length of 7 characters, this colour is first transformed to decimal values, by *dec2hex.pl*, and then converted from RGB to HSL values by *rgbtoshl.pl*.

rgbtoshl.pl

The *rgbtoshl* module converts RGB encoded colours to HSL encoded colours. Input values for R, G and B are in the range from 0 to 255 and output values for H are in the range from 0 to 359 and for S and L in the range from 0 to 255. The conversion is described in appendix 3.

dec2hex.pl

This module is used by *rgbtoshl* and *hsltorgb* to convert decimal values to hexadecimal values and vice versa. Some of the parameters are described below.

config.pl

This is somewhat different from the files described previously. This prolog module contains parameter values that are used in the decision process and can thus be used to *adjust* the system.

wc The parameter values for wc stand for warm/cool and are used in ranking foreground/background combinations. We distinguish between positive, neutral and negative parameters with values 1.2, 0.9 and 0.4 respectively.

ld These parameter values stands for light/dark and are also used in ranking foreground/background combinations. We distinguish between positive and negative results with values of 1.2 and 0.6 respectively.

coollimit Here the upper and lower limits of warm and cool colours are set. Parameters are thus lower and upper with value of 120 and 330. These values are used in different modules of the system.

skim The only parameter currently in skim is “purity” with a value of 0.9. This value is used to determine the purity of a colour—the proximity to the outer ring of HSL colour space. A colour with HSL values [0, 255, 128] is considered as a pure colour and obtains a value of 1, which is calculated by $\frac{\text{saturation}}{255} \times \left(1 - \frac{128 - \text{lightness}}{255}\right)$. If this value is above the value of “purity” we consider this colour to be a *relatively* pure spectral colour. This value is used for ranking foreground/background combinations with complementary colours, which has to be avoided when they are pure spectral colours.

reusebgcolour This parameter can be set to **true** or **false** and tells the system to (dis)allow reusing background colours for the different areas.

steps This parameter provides the stepsize for labelling the suggested colours. The value is set to 5 because colours [hue, 255, 128] and [hue +/- steps, 255, 128] are perceived as similar.

Having described the different modules the Design Module is composed of and the framework in which it resides we are now able to test and evaluate the Design Module with different instantiated Constraint_lists and scenarios.

2.2 Testing and evaluation

The “De Stijl” example scenario is a good example of how the system decides to pick and suggest colours. By adjusting some input parameters we are able to see if the system still does the right job. Figure 6.5 shows us 4 examples where input parameters of the “De Stijl” scenario have been altered:

- 1 This example is the original scenario, where red, blue and yellow are the domain colours, colour blindness is set to red_green and the form function balance is set to *function*.
- 2 In example 2 the form function balance is set to *form*. The most critical change can be seen in content_area/1, because the red on yellow combination now exceeds the (lower) threshold, this combination is accepted. *Assignlinks* decides that reusing the link colour of content_area/1 isn’t sufficient when assigning link_area/2 with the background colours and calls the *suggestor*, which proposes orange on blue. This is accepted and the area on the left now has a blue background.
- 3 In example 3 we use the same settings as the original scenario, but omitted two of the domain colours, namely yellow and red. Because the weight of the area together with the form function balance is set to *function* the system decides for black on white for content_area1. The remaining colour is blue and the system decides to use



(a) Example 1



(b) Example 2



(c) Example 3



(d) Example 4

Figure 6.5: “De Stijl” page in 4 different output examples

this colour for the links. The link_area2 is assigned blue as a background colour with black as the link colour. When assigning colours to link_area/1 the system decided by using the flipper method.

- 4 In example 4 we set the form function balance to *neutral*, omitted the domaincolours yellow and red and added cyan and cyanblue as preferred colours. The result is a nice analogue cool scheme, in which legibility factors are utilized in an acceptable way.

To show that the system is not only capable of working with the “De Stijl” example scenario, we used a second example scenario, partially based on a presentation generated by the Cuypers system. Figure 6.6 shows a screenshot of a presentation generated for the style “Genre Painting” and the subject “Johannes Vermeer”¹.



Figure 6.6: Screenshot of an automatically generated SMIL presentation

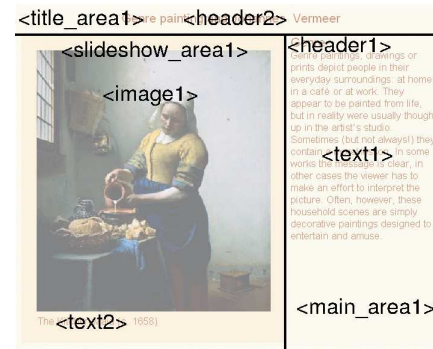


Figure 6.7: Graphical representation of the spatial layout

The page generated consists of three areas, as presented in Figure 6.7. The top area, containing the title, we call title_area1. The area on the left, containing the painting from “Johannes Vermeer” we call slideshow_area1 because in the generated SMIL presentation this area shows different images in a sequence. Finally we can distinguish main_area1 which contains the text about the subject “Genre Painting”.

The original colour design has been created by hand and consists of three colours. Colour one is the background colour of slideshow_area1 and has the HSL values [33,255,224]. Colour two is the colour of the text from main_area1 and is HSL-coded by [0,255,49]. The last colour is the background colour for main_area1 and title_area1 and this is in HSL[39,217,242], these colours were originally coded in RGB values, see also Figure 6.8.




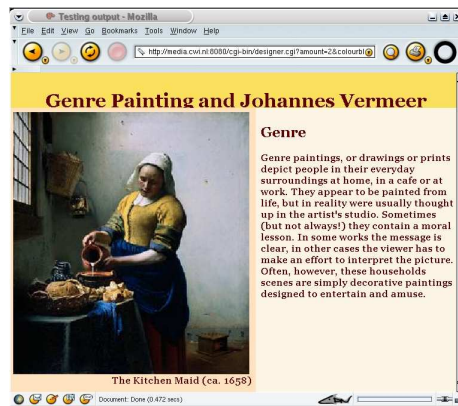
	Colour 1 #ffe4c4 hsl(33,255,224)
	Colour 2 #600000 hsl(0,255,49)
	Colour 3 #fdf5e6 hsl(39,27,242)

Figure 6.8: The domain colours in RGB and HSL, supplied to the “Vermeer” scenario.

We supplied these three colours as domain colours to the system. Using these colours the *subjectanalyzer* comes up with an analogue scheme. The *processconstraintlist* module decides about the order in which the areas need colour assignment, based on the form-function balance set to *function*. The result is main_area1, slideshow_area1 and finally title_area1. For this particular example we will also describe four examples, which can be seen in Figure 6.9.

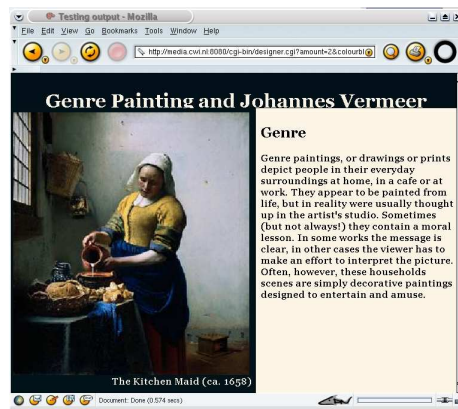
¹The painting is called “The Kitchenmaid” by Johannes Vermeer (1658) and is taken from [1].



(a) Example 1



(b) Example 2



(c) Example 3



(d) Example 4

Figure 6.9: "Genre Painting and Johannes Vermeer" page in 4 different scenarios

1. Example 1 is the result we get by using only the 3 domain colours and the form function balance set to *function*. The lightest and darkest colour are approved by the system to be used as foreground and background colour combination.
2. In example 2 we added white as a preferred colour, resulting in a *nice* harmonious analogue page. The system decides that colour 2 on white is a sufficient legible combination and use these for mainarea/1. Colour 2 is used for the background of slideshowarea/1 and colour 3 is used for the background of titlearea/1.
3. In example 3 we let the Design Module use only a single domain colour, by omitting colours 1 and 2. This is a test to show the system is able to generate colours instead of only picking colours from a pool of provided colours. As we only provide one colour, the system is able to use more schemes than the analogue scheme. As form function balance is set to *function* the system decides black is the best foreground colour on the provided domain colour for mainarea/1. Slideshowarea/1 obtains a complementary scheme where the background colour is the complementary for the foreground colour, which is the same as the provided domain colour. As this domain colour is very bright, the background and thus complementary colour is very dark, for the trained eye this colour is different from black (hsl[195, 217, 13]). The combination for titlearea/1 is similar to the one as for the slideshowarea, except that the background colour is in HSL coded as [200,217,13], which is a not noticeable difference.
4. Example 4 shows the result if the form function balance is set to *function* and an additional colour is chosen, namely cyan. Only the titlearea is influenced by this decision as cyan is used here as the background colour. This is not an optimal result and needs adjustment.

The above 8 examples from 2 scenarios show that the Design Module is able to pick and suggest colours in a way that fits the decided upon harmonious scheme. Note: aesthetics are not to be evaluated in these examples. We only tested whether the system does what it is supposed to do regarding its requirements. The presented results describe that the system takes decisions it was allowed to do, and expected to do and thus fits the requirements.

3. INTEGRATION

To make use of the systems capabilities it is necessary to integrate the system in an environment in which its capabilities can be exploited. The creation of the Design Module was done standalone, because of the ease of testing and debugging during the realisation phase. After we had made progress, however, it was necessary to be able to see the actual colours the system used and suggested. A web-interface was the perfect solution to realise a quick integration. The logical consequence after building the web-interface was the integration into the Cuypers system.

3.1 Web Environment

When using a web interface it is possible to directly see the results of the input parameters. We defined the two scenarios described above and made the parameters adjustable. Figure 6.10 displays the webinterface².

Input parameters include the number of the colours that should be used, which was at disabled at the time of writing, a colour-blindness checkbox and the choice between the three options for the form-function balance. The next section of the interface is the selection of preferred colours the user wants to use. The textbox provides a way of entering HSL coded colours. The user is then able to select between the “De Stijl” or the “Genre Painting and Vermeer” example scenario. The domain colours are optional, but by default checked. The final option is the choice between the output page, as in Figure 6.5 on page 56 and as

²This interface can be found at <http://media.cwi.nl/~media/demo>.

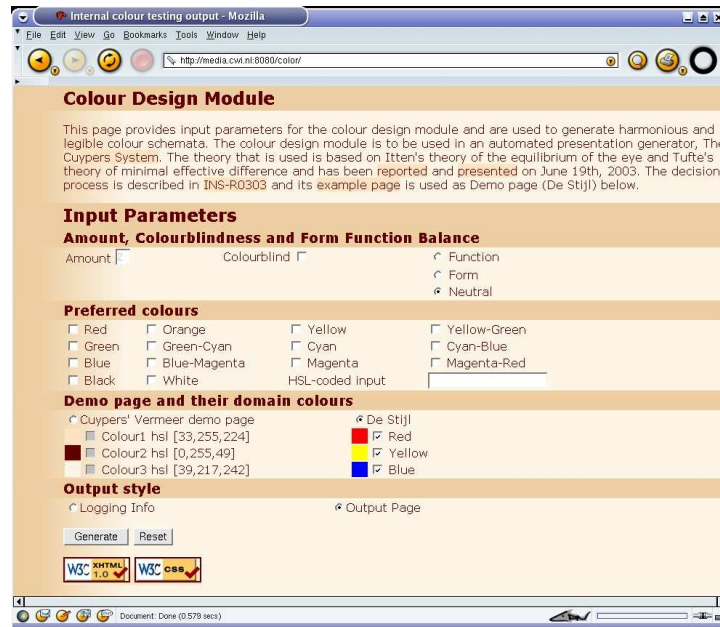


Figure 6.10: The web interface

in Figure 6.9 on page 58, and a log page. The log page is mainly used for investigating the Design Module's decisions.

When pressing *submit* a “cgi-script” is used to generate a command which is used to call “eclipse” with reference to the top level module from the Design Module *designer* and the correct parameters. Figure 6.11 shows the command that is used with the generation of the example in Figure 6.5(a). In Figure 6.4 on page 53 we see a similar call entering the Design Module.

```
/usr/local/bin/eclipse -e
"['../../color/designer.pl'],
main([red,yellow,blue],[],[],2,green_red,function,destijl,page)."
```

Figure 6.11: The command generated by the “cgi-script”

The last parameter in this call instructs the toplevel handler to make a call to a utility module, named *createhtml*, which writes out XHTML with included CSS, which represents the layout of the page and the style attributes. This output is fetched by the “cgi-script” which by in turn presents this XHTML and CSS in the web browser with “Content-type” set to “text/html”. Figure 6.4 on page 53 shows the XHTML+CSS output the Design Module.

3.2 Cuypers

The integration into the Cuypers System meant determining the right code placement of the Design Module, in contrast to the web interface, where a frontend and a backend needed to be built.

Chapter 2 described the separation between content, presentation structure and style. The Cuypers Presentation Engine (CPE) uses a specific multimedia formatting vocabulary to achieve this. Thus, before we describe the integration of the Design Module into the Cuypers system, we first provide a brief overview of the CPE vocabulary.

*Hypermedia Formatting Objects*³ When using SMIL, or any other form of XML-based multimedia presentations, we need the same functionality as we are used to when using XML-based text document transformations with XSLT. Requirements for multimedia styling and a formatting vocabulary are defined for:

- Visual layout
- Visual style
- Temporal structure
- Temporal style
- Support for top-down transformations
- Support for bottom-up transformations
- Abstraction from target output format
- Support for resource-constrained presentations
- Support for generating consistent presentations

For styling multimedia presentations, extra functionality and extended CSS is needed, such as timing and transition style properties. A small set of multimedia formatting objects have been implemented in the CPE. These are focussed on constrained layouts and temporal behaviour. Different formatting objects are described in an example scenario in [56]. These formatting objects are called Hypermedia Formatting Objects (hfos). An example of an hfo is the **hfo:image**. This object models an image and contains contextual information about the image. Information included might be height, width, URL etc., but can also contain preferred maximum and minimum duration and metadata information. Other hfo objects are `hfo:text`, `hfo:vbox`, `hfo:hbox`, `hfo:slideshow` and `hfo:root`.

Integration of the Design Module in Cuypers Returning to the Design Module and its integration within Cuypers, we can state that the assignment of the colours to the areas and their elements needs to be done after obtaining sufficient knowledge about the presentation to be generated. This means, for example, knowing the different areas and their elements that are to be used during the generation of the presentation. During the generation of multimedia presentations by the CPE, we can distinguish between two major transformation steps, when dealing with ontology driven generation [16]. The first step, the transformation from the semantic graph to a structured progression, is out of scope of the current work, the second step is a transformation from the structured progression to a multimedia presentation, which the Design Module supports. In this transformation step the various areas and elements are known and are passed to the module which creates the hfos. This transformation step is where the Design Module is integrated in the CPE. We are using the data from the presentation structure to fill in the layout and domain aspects for the `ConstraintList`, which are fed into the Design Module. The colours returned from the Design Module are then used to create the hfos⁴.

4. CONCLUSION

This chapter provided an insight into the realisation of the Design Module and its architecture. The established research theory, provided by chapters 2, 3 and 4, helped in designing an example scenario. This example scenario was used during the realisation of the Design Module, one of the modules attached to the Cuypers Presentation Engine (CPE). Integration of the code for the Design Module was necessary to allow proper evaluation of the

³based on [56].

⁴The Design Module, integrated in the Cuypers Engine can be approached from <http://www.cwi.nl/~media/demo>.

performance of the colour picking process. We decided first for a web interface, because of the ease of testing. After a sufficiently long testing period, the integration of the code into the CPE was performed.

Colour decisions made by the system, with respect to the system requirements, are implemented according to the established theories. Aesthetics on the overall colour design, however, needs further evaluation. Certain colour decisions by the system provide a harmonious, yet unfortunately unattractive presentation.

Chapter 7

Conclusion

This thesis has presented a rule based approach to the automatic generation of correct colour schemes, to convey the communicational goal of a presentation. We started this work with an extensive analysis on theory regarding web documents in the context of automated hypermedia presentation generation. This included topics such as evolution of web documents and the inevitable separation of content from style. We continued our analysis with a detailed description of the automated presentation generator Cuypers. By using an example scenario the presentation generation process within the Cuypers system was explained. This provided an overview of the environment our Design Module resides in.

The derivation of colour theory with respect to physics, perception and cognitics, set a fundament, with the use of theories from Itten [21] and Tufte [50, 51], for defining colour harmony. Typography focussed on colours and legibility, was investigated and this provided guidelines for graphic design. By combining colour harmony and legibility factors we created general rules which can be applied in graphic design and can be used in design decisions within the automatic generation of hypermedia presentations.

By means of an example scenario we propose an approach to automate the colour design, in which we take the created graphic design rules, the user requirements, the domain characteristics and the presentation characteristics into consideration. We allow the Design Module to support a formal (aesthetics driven) or a functional (legibility driven) generation of multimedia presentations.

By realising the Design Module, we obtained insights in colour design and the applicability of the colour harmony theories. We demonstrate the flexibility and robustness of the system by adapting the user requirements or domain characteristics in the provided and an additional test scenario.

Integration of the automatic colour design in the Cuypers system and in a web interface gave the opportunity to see the results the system generates. Testing and evaluation showed the system was capable of generating, depending on the constraint set, user adapted pages with overall acceptable colour representations. We provide with this work a modular framework in which the Design Module resides. This framework can be seen as the starting point of automated colour design to convey the communicational goal, taking the style attributes of the constituent media elements into account.

1. EVALUATION

Due to the dynamic environment in which Cuypers resides, in which neither the user requirements nor the requested material can be predicted in advance, we have developed the first dynamic automated colour design mechanism. The uniqueness of the Design Module lies within its dynamic aspects, in contrast to [27], for example.

By doing extensive research on colours and legibility we converted implicit design knowledge into explicit, usable knowledge in the form of general graphic design rules. These relevant design aspects of colour theory have been integrated in the automated colour design mechanism, the Design Module. By realising the Design Module we were able to test and evaluate these rules. Research on legibility factors for displays and general legibility constraints provided knowledge about background foreground combinations. Using Itten's theory based on the equilibrium of the eye tends the Design Module to come up with a more variegated presentation. Tufte's minimal effective difference theory came up with less variegated colour schemes because of the milder differences between colours. Aesthetics with respect to colour harmony have been, according to the different schemes from section 2.5, inserted in the Design Module. This means harmony is achieved within the scope of the definition of colour harmony schemes. These schemes provide the Design Module with a variety of possibilities in assigning colours to areas and their elements. Certain schemes may need, however reconsideration, after a possible evaluation-study. For example the hexad scheme, which provides the system with the possibility in choosing 6 different colours, may cause a too variegated presentation.

With respect to form-function balance, the schemes provide sufficient flexibility to allow a more *aesthetical* presentation, or to allow a more *functional* presentation or a presentation in between these two. Conveying the communicational goal with the help of this form function balance resulted in a research question of how to allow the system to decide autonomously when to overrule a functional decision with a formal solution or the other way round because the result serves the current communicational goal best. Mechanisms have been designed and integrated within the Design Module. For example the possibility to let the system alter a colour scheme, when the communicational goal cannot be achieved with the current set of colours. Another example is the *formal* "flipper" method which can be used within a "functional" communicational goal, by introducing weights to different areas and elements.

The overall architecture of the Design Module is considered to be a flexible, modular framework which provides a fundament on which further development can be done. This modular quality of the architecture allows independently altering or even reprogramming the different components.

One has to keep in mind that we do not claim to produce optimal colour schemes for the automated generation of presentations, we attempted, nevertheless, by balancing between form and function to optimise conveying the communicational goal.

2. FINAL REMARKS AND FUTURE WORK

We are aware of the fact that some components needs further elaboration with respect to detail. For example the expression of the various states and functionalities of a link through colour [62, 24], the relation of colour and other visual information units and more variations on how to choose between colour schemes.

Further research is required on the relation between the background colour and the foreground colour of an image. Approaches may vary from taking an *average* colour based on a colour histogram, or the average colour of the outer image boundary. Recalling the "Genre Painting and Johannes Vermeer" example scenario, where the images were placed on a slideshow area, an interesting point of research would be letting the temporal attributes of the hfo:slideshow object influence the average colour, used for these images. Another topic which may need further research is the actual area a particular colour occupies, resulting in the colourpicker taking also the areaisize into account when assigning colours to areas or their elements.

Furthermore, we need to investigate clashing colours, like the combination of pink and orange, and, although part of the Design Module but not investigated yet, legibility factors with respect to typography.

Interesting research questions still remain, such as how to adapt our environment to work with an additional virtual colour space that contains those potential colours that might be presented at a later stage, because particular functionality of an information unit might not be required all the time. Another research question is how valid our hypothesis is that the automated generation of dynamic presentations can be improved by providing a balance between form and function.

Appendix I

Conversion

1. HSL \leftrightarrow YIQ

$$\begin{pmatrix} Y \\ I \\ Q \end{pmatrix} = \begin{pmatrix} 0.299 & 0.587 & 0.114 \\ 0.596 & -0.275 & -0.321 \\ 0.212 & -0.532 & 0.311 \end{pmatrix} \times \begin{pmatrix} R \\ G \\ B \end{pmatrix}$$

2. HSL \rightarrow RGB

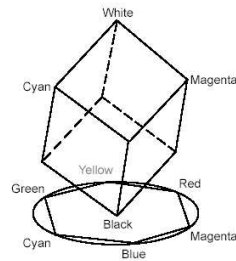


Figure I.1: Visualization of conversion from RGB to HSL

The conversion from HSL to RGB and vice-versa can be visualized with Figure I.1. With the use of ABC, see [18], the conversion is described below, this conversion is originally from [13].

```
HOW TO RETURN hsl.to.rgb(h, s, l):
SELECT:
  l<=0.5: PUT l*(s+1) IN m2
  ELSE: PUT l+s-l*s IN m2
PUT l*2-m2 IN m1
PUT hue.to.rgb(m1, m2, h+1/3) IN r
PUT hue.to.rgb(m1, m2, h) IN g
PUT hue.to.rgb(m1, m2, h-1/3) IN b
RETURN (r, g, b)
```

```

HOW TO RETURN hue.to.rgb(m1, m2, h):
  IF h<0: PUT h+1 IN h
  IF h>1: PUT h-1 IN h
  IF h*6<1: RETURN m1+(m2-m1)*h*6
  IF h*2<1: RETURN m2
  IF h*3<2: RETURN m1+(m2-m1)*(2/3-h)*6
  RETURN m1

```

3. RGB → HSL

```

HOW TO RETURN rgb.to.hsl(r, g, b):
  PUT get.max3(r, g, b) IN max
  PUT get.min3(r, g, b) IN min
  PUT (max + min)/2 IN l
  IF max = min: PUT 0 IN s
                  PUT 0 IN h
  SELECT:
    l<0.5: PUT (max-min)/(max+min) IN s
    ELSE: PUT (max-min)/(2.0-max-min) IN s
  IF r=max: PUT (g-b)/(max-min) IN h
  IF r=max: PUT 2.0 + (b-r)/(max-min) IN h
  IF r=max: PUT 4.0 + (r-g)/(max-min) IN h
  RETURN (h, s, l)

```

```

HOW TO RETURN get.max3(r, g, b):
  PUT get.max2(r,g) IN t
  PUT get.max2(t,b) IN max
  RETURN max

```

```

HOW TO RETURN get.max2(x, y):
  SELECT:
    x <= y: RETURN y
    ELSE: RETURN x

```

```

HOW TO RETURN get.min3(r, g, b):
  PUT get.min2(r,g) IN t
  PUT get.min2(t,b) IN min
  RETURN min

```

```

HOW TO RETURN get.min2(x, y):
  SELECT:
    x <= y: RETURN x
    ELSE: RETURN y

```

4. CMYK ↔ RGB

$$\begin{pmatrix} C \\ M \\ Y \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} - \begin{pmatrix} R \\ G \\ B \end{pmatrix}$$

Appendix II

Acronyms

ARPANET	Advanced Research Projects NETwork
ASP	Active Server Page
BGN	Boekhandels Groep Nederland
CC/PP	Composite Capabilities/Preference Profiles
CIE	Commission International de L'Eclairage
CMYK	Cyan Magenta Yellow black
CPE	Cuypers Presentation Engine
CSS	Cascading StyleSheets
DBMS	DataBase Management System
hfo	Hypermedia Formatting Object
HSL	Hue Saturation Lightness
HSV	Hue Saturation Value
HTML	HyperText Markup Language
IP	Internet Protocol
ISA	Information Systems Algorithms
ITS	Information Technology and Systems
NCSA	National Centre for Supercomputing Applications
nm	nanometer
NSFNET	National Science Foundation NETwork
OWL	Web Ontology Language

RDF Resource Description Framework

RDFS RDF Schema

RGB Red Green Blue

RST Rhetorical Structure Theory

SMIL Synchronized Multimedia Integration Language

SQL Structured Query Language

TCP Transmission Control Protocol

TU-Delft Technical University Delft

URI Uniform Resource Identifier

URL Uniform Resource Locator

WWW World Wide Web

XHTML eXtensible Hypertext Markup Language

XML eXtensible Markup Language

XSL eXtensible Stylesheet Language

XSLT XSL Transform

XSP eXtensible Server Page

References

1. R. Amsterdam. Rijksmuseum Amsterdam Website. <http://www.rijksmuseum.nl>.
2. E. André, J. Müller, and T. Rist. *WIP/PPP: Knowledge-Based Methods for Fully Automated Multimedia Authoring*. London, UK, 1996.
3. J. Bateman, J. Kleinz, T. Kamps, and K. Reichenberger. Towards Constructive Text, Diagram, and Layout Generation for Information Presentation. *Computational Linguistics*, 27(3):409–449, September 2001.
4. D. Bauer and C. Cavonius. *Ergonomic aspects of visual display terminals*, chapter Improving the legibility of visual display units through contrast reversal, pages 137–142. Taylor and Francis, 1980.
5. U. Baumann, N. Silverstrini, and E. P. Fischer. Colour Museum. <http://www.colorsystem.com/index.htm>.
6. D. Baylor. *Colour: Art and Science*, chapter Colour Mechanisms of the Eye, pages 103–126. Cambridge University Press, 1999.
7. T. Berners-Lee. *Weaving the Web*. Orion Business, 1999.
8. S. Boll, W. Klas, and J. Wandel. A Cross-Media Adaptation Strategy for Multimedia Presentations. In *ACM Multimedia '99 Proceedings*, pages 37–46, Orlando, Florida, October 30 - November 5, 1999. ACM, Addison Wesley Longman.
9. P. Brusilovsky. Adaptive Hypermedia. *Journal on User Modeling and User-Adapted Interaction*, 11:87–110, 2001.
10. H. Cooper and R. Spronk. *Mondrian: The Transatlantic Paintings*. Yale University Press, 2001.
11. K. Czajka. Design of interactive and adaptive interfaces to exploit large media-based knowledge spaces in the domain of museums for the fine arts. Diploma Media System Design, University of Applied Sciences Darmstadt, June 14, 2002.
12. R. den Buurman. Use and abuse of color in designing for computer displays. In *Proceedings of Interacting with Computers Preparing for the Nineties*, pages 185–188. Stichting Informatica Congressen, Amsterdam, December 1994.
13. J. D. Foley. *Fundamentals of Interactive Computer Graphics*. Addison-Wesley Pub Co, 1982.
14. M. M. G. Davenport. ConText: Towards the Evolving Documentary. In *ACM Multimedia '95, Proceedings*, pages 377–378, November 1995.

15. J. Geurts. Constraints for Multimedia Presentation Generation. Master's thesis, University of Amsterdam, 2002.
16. J. Geurts, S. Bocconi, J. van Ossenbruggen, and L. Hardman. Towards Ontology-driven Discourse: From Semantic Graphs to Multimedia Presentations. In *Second International Semantic Web Conference (ISWC2003)*, Sanibel Island, Florida, USA, October 20-23, 2003. To be published.
17. J. Geurts, J. van Ossenbruggen, and L. Hardman. Application-Specific Constraints for Multimedia Presentation Generation. Technical Report INS-R0107, CWI, May 31, 2001.
18. L. Geurts, L. Meertens, and S. Pemberton. *The ABC Programmer's Handbook*. Prentice-Hall, 1991.
19. Guggenheim museum. Guggenheim Museum - The Collection. <http://www.guggenheimcollection.org>.
20. E. Heller. *Wie Farben auf Gefühl und Verstand wirken*. Droemer Verlag, Munchen, 2000.
21. J. Itten. *The Art of Color*. John Wiley and Sons, Inc., 2002.
22. T. Kamps. *Diagram Design : A Constructive Theory*. Springer Verlag, 1999.
23. M. Kohnke. A Sense of Type. http://www.weassociated.com/A_Sense_of_Type/index.html.
24. H. W. Lie, B. Bos, and R. Caillau. *Cascading Style Sheets: Designing for the Web (2nd Edition)*. Addison-Wesley Pub Co, 1999.
25. S. Little. Cuypers Meets Users: Implementing a User Model Architecture for Multimedia Presentation Generation. To be published as CWI-Techreport, 2002.
26. M. Longair. *Colour: Art and Science*, chapter Light and Colour, pages 65–102. Cambridge University Press, 1999.
27. P. Lyons, G. Moretti, and M. Wilson. Colour Group Selection for Computer Interfaces. In B. Rogowitz and T. Pappas, editors, *Human Vision and Electronic Imaging V*, volume 3959, pages 302–313, San Jose, USA.
28. B. MacEvoy. Handprint: Color, 2002. <http://www.handprint.com/HP/WCL/wcolor.html>.
29. W. C. Mann, C. M. I. M. Matthiesen, and S. A. Thompson. Rhetorical Structure Theory and Text Analysis. Technical Report ISI/RR-89-242, Information Sciences Institute, University of Southern California, November 1989.
30. J. Morton. Color Matters, 2002. <http://www.colormatters.com/>.
31. F. Nack. Pictures at an exhibition. *IEEE Multimedia*, pages 14–17, April - June 2002.
32. F. Nack and L. Hardman. Denotative and Connotative Semantics in Hypermedia: Proposal for a Semiotic-Aware Architecture. *New Review of Hypermedia and Multimedia*, 7:7–37, 2001.
33. F. Nack, A. Manniesing, and L. Hardman. Colour Picking, the Pecking Order of Form and Function. Technical Report INS-R0303, CWI, June 2003.
34. F. Nack, M. Windhouwer, L. Hardman, E. Pauwels, and M. Huijberts. The Role of High-level and Low-level Features in Style-based Retrieval and Generation of Multimedia Presentations. *New Review of Hypermedia and Multimedia*, 7:39–65, 2001.
35. Y. Ohno. CIE Fundamentals for Color Measurement. *IS&T NIP16*, October 2000.
36. G. S. Owen. Physiological Principles for the Effective Use of Color. Online, 1999. based on "Physiological Principles for the Effective Use of Color", G. Murch, IEEE,

- pp. 49-54, Nov., 1984.
37. N. Poppelier and J. Braams. A style option to adapt the standard LaTeX document styles to A4 paper. May 1997.
 38. M. Roberts. The CIE Colour System.
<http://www.cs.bham.ac.uk/~mer/colour/cie.html>.
 39. Q. Roper. De Stijl and the Modern Movement.
http://www.qdesign.co.nz/designhist_destijl.html.
 40. L. Rutledge, B. Bailey, J. van Ossenbruggen, L. Hardman, and J. Geurts. Generating Presentation Constraints from Rhetorical Structure. In *Proceedings of the 11th ACM conference on Hypertext and Hypermedia*, pages 19–28, San Antonio, Texas, USA, May 30 – June 3, 2000. ACM.
 41. L. Rutledge, J. Davis, J. van Ossenbruggen, and L. Hardman. Inter-dimensional Hypermedia Communicative Devices for Rhetorical Structure. In *Proceedings of the International Conference on Multimedia Modeling 2000 (MMM00)*, pages 89–105, Nagano, Japan, November 13-15, 2000.
 42. L. Rutledge, J. van Ossenbruggen, L. Hardman, and D. C. Bulterman. Mix'n'Match: Exchangeable Modules of Hypermedia Style. In *Proceedings of the 10th ACM conference on Hypertext and Hypermedia*, pages 179–188, Darmstadt, Germany, February 21-25, 1999. ACM. Edited by Klaus Tochtermann, Jorg Westbomke, Uffe K. Will and John J. Leggett.
 43. P. Schmitz, J. Yu, and P. Santangeli. Timed Interactive Multimedia Extensions for HTML (HTML+TIME): Extending SMIL into the Web Browser. W3C Note are available at <http://www.w3.org/TR/1998/NOTE-HTMLplusTIME-19980918>, September 1998.
 44. K. Schwarz. An investigation on the relationship between the user model and graphic representations for the automated generation of multimedia presentations. Diploma Media System Design, University of Applied Sciences Darmstadt, 2003.
 45. M. Q. Shaw. Evaluating the 1931 CIE Color Matching Functions. Master's thesis, University of Hertfordshire, England, 1997.
 46. H. L. Snyder, J. J. Decker, C. J. Lloyd, and C. Dye. Effect of the Image Polarity on VDT Task Performance. In *Proceedings of the Human Factors Society*, 1990.
 47. M. Spruit. Management van Informatiesystemen, 2002. TU-Delft dictaat bij het vak Informatiestrategie en Beheer van Informatiesystemen - In3020.
 48. D. P. Stern and M. Perede. The Exploration of the Earth's Magnetosphere, January 2003. <http://www-istp.gsfc.nasa.gov/Education/wmap.html>.
 49. A. S. Tanenbaum. *Computernetwerken*. Academic Service, Schoonhoven, 1998. Copublication from Prentice Hall International, Dutch translation by L. Geurts, Original Title: Computer Networks, Third Edition.
 50. E. R. Tufte. *Envisioning Information*. Graphics Press, 1999.
 51. E. R. Tufte. *Visual Explanations*. Graphics Press, 2000.
 52. Dictionary.com. <http://www.dictionary.com>.
 53. K. van der Meer. *Documentaire Informatiesystemen*. NBLC Uitgeverij, third revised edition, 1998.
 54. J. van Ossenbruggen. *Processing Structured Hypermedia — A Matter of Style*. PhD thesis, Vrije Universiteit, Amsterdam, The Netherlands, April 10, 2001.
 55. J. van Ossenbruggen, J. Geurts, F. Cornelissen, L. Rutledge, and L. Hardman. Towards Second and Third Generation Web-Based Multimedia. In *The Tenth International World Wide Web Conference*, pages 479–488, Hong Kong, May 1-5,

2001. IW3C2, ACM Press.
56. J. van Ossenbruggen, J. Geurts, L. Hardman, and L. Rutledge. Towards a Formatting Vocabulary for Time-based Hypermedia. In *The Twelfth International World Wide Web Conference*, pages 384–393, Budapest, Hungary, May 20–24, 2003. IW3C2, ACM Press.
 57. J. van Ossenbruggen and L. Hardman. Smart Style on the Semantic Web. In *Semantic Web Workshop, WWW2002*, May 2002.
 58. J. van Ossenbruggen and L. Hardman. Multimedia on the Semantic Web. In H. van Oostendorp, A. Dillon, and L. Breure, editors, *Creation, Use and Deployment of Digital Information*. Erlbaum, Publication planned late 2003.
 59. W3C. Synchronized Multimedia Integration Language (SMIL) 1.0 Specification. W3C Recommendation, June 15, 1998. Edited by Philipp Hoschka.
 60. W3C. Composite Capability/Preference Profiles (CC/PP): Structure and Vocabularies. Work in progress. W3C Working Drafts are available at <http://www.w3.org/TR>, 15 March 2001. Edited by Graham Klyne, Franklin Reynolds, Chris Woodrow and Hidetaka Ohto.
 61. W3C. Synchronized Multimedia Integration Language (SMIL 2.0) Specification. W3C Recommendation, August 7, 2001. Edited by Aaron Cohen.
 62. H. Weinreich, H. Obendorf, and W. Lamersdorf. The Look of the Link — Concepts for the User Interface of Extended Hyperlinks. In *Proceedings of the 12th ACM conference on Hypertext and Hypermedia (HT01)*, pages 19–28, University of Aarhus, Århus, Denmark, August 14–18, 2001.
 63. L. Weitzman and K. Wittenburg. Automatic presentation of multimedia documents using relational grammars. In *Proceedings of the second ACM international conference on Multimedia '94*, pages 443–451, San Francisco, October 15 - 20, 1994.
 64. T. G. Wolfmaier. Designing for the Color-Challenged: A Challenge. *Internetworking*, march 1999.
http://www.internettg.org/newsletter/mar99/accessibility_color_challenged.html.
 65. S. Wu. The Psychology of Color, 2003.
<http://psychology.about.com/library/weekly/aa031501a.htm>.
 66. K.-W. Yau. Visual and olfactory transductions - a tale of two senses, July 2000.
<http://physiology.cup.cam.ac.uk/Proceedings/Abstracts/527P/Cambridge/Files/sympS10.html>.