List of Todos

Go to TOC

Go to TOC

Go to TOC

Go to TOC

Institute of Creative Technologies
De Montfort University

FANIA RACZINSKI

ALGORITHMIC META-CREATIVITY

Creative Computing and Pataphysics for Computational Creativity

pata.physics.wtf

Supervisors:
Prof. Hongji YANG
Prof. Andrew HUGILL
Dr. Sophy SMITH
Prof. Jim HENDLER

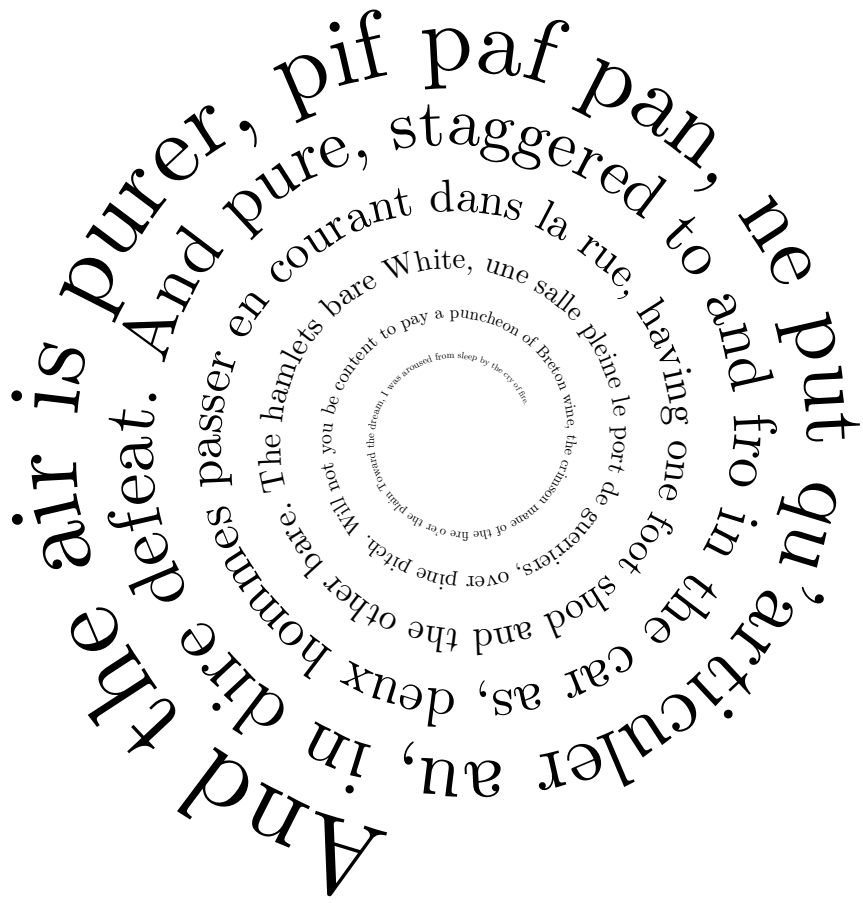
A thesis submitted in partial fulfilment of the requirements for the degree of Doctor of Philosophy

Created: 25th March 2015 — Last Saved: 12th August 2016
Wordcount:

17983 (errors:29)

Go to TOC

PRE☺


And the air is purer, pif paf pan, ne put qu'article au, in dire defeat. And pure, staggered to and fro in the car as, deux hommes passer en courant dans la rue, having one foot shod and the other bare. The hamlets bare White, une salle pleine le port de Guetteurs, over pine pitch. Will not you be content to pay a puncheon of Breton wine, the cr.timson mantle of the fire o'er the plain. Toward the dream, I was aroused from sleep by the cry of fire.

Go to TOC

TL;DR

Algorithmic Meta-Creativity — Fania Raczinski — Abstract[1]

Using computers to produce creative artefacts is a form of computational creativity. Using creative techniques computationally is creative computing. Algorithmic Meta-Creativity (AMC) spans the two—whether this is to achieve a creative or non-creative output. It is the use of digital tools (which may not be creative themselves) and the way they are used forms the creative process or product. Creativity in humans needs to be interpreted differently to machines. Humans and machines differ in many ways, we have different 'brains/memory', 'thinking processes/software' and 'bodies/hardware'. Too often creative output by machines is judged as we would a humans. Computers which are truly artificially intelligent might be capable of true artificial creativity. Until then they are (philosophical) zombie robots: machines that behave like humans but aren't conscious. The only alternative is to see any computer creativity as a direct or indirect expression of human creativity using digital means and evaluate it as such. AMC is neither machine creativity nor human creativity—it is both. By acknowledging the undeniable link between computer creativity and its human influence (the machine is just a tool for the human) we enter a new realm of thought. How is AMC defined and evaluated? This thesis address this issue. First a practical demonstration of AMC is presented (`pata.physics.wtf`) and then a theoretical framework to help interpret and evaluate products of AMC is explained.

Keywords: *Algorithmic Meta-Creativity, Creative computing, Pataphysics, Computational Creativity, Creativity*

[1] "Too long; didn't read"

PUBLICATIONS

Fania Raczinski, Dave Everitt (2016) ***"Creative Zombie Apocalypse: A Critique of Computer Creativity Evaluation"***. Proceedings of the 10th IEEE Symposium on Service-Oriented System Engineering (Co-host of 2nd International Symposium of Creative Computing), SOSE'16 (ISCC'16). Oxford, UK. Pages 270–276.

Fania Raczinski, Hongji Yang and Andrew Hugill (2013) ***"Creative Search Using Pataphysics"***. Proceedings of the 9th ACM Conference on Creativity and Cognition, CC'13. Sydney, Australia. Pages 274–280.

Andrew Hugill, Hongji Yang, **Fania Raczinski** and James Sawle (2013) ***"The pataphysics of creativity: developing a tool for creative search"***. Routledge: Digital Creativity, Volume 24, Issue 3. Pages 237–251.

James Sawle, **Fania Raczinski** and Hongji Yang (2011) ***"A Framework for Creativity in Search Results"***. The 3rd International Conference on Creative Content Technologies, CONTENT'11. Rome, Italy. Pages 54–57.

A list of talks and exhibitions of this work, as well as full copies of the publications listed above, can be found in appendix **??**.

Contents

Go to TOC

Figures

Tables

Code

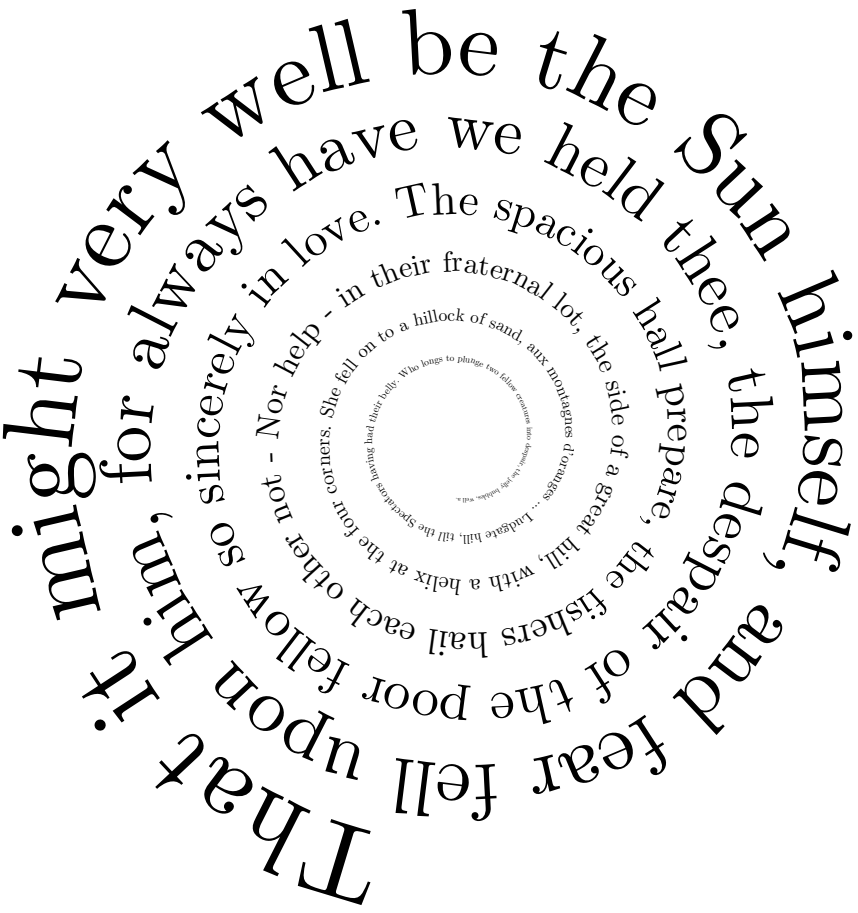
Go to TOC

Acronyms

AMC	Algorithmic Meta-Creativity
IR	Information Retrieval
NLP	Natural Language Processing
AI	Artificial Intelligence
NLTK	Natural Language Tool Kit
TDM	Term-Document Matrix
API	Application Program Interface
REST	Representational State Transfer
HTTP	Hypertext Transfer Protocol
RDF	Resource Description Framework
URL	Uniform Resource Locator
JSON	JavaScript Object Notation
HTML	Hypertext Markup Language
CSS	Cascading Stylesheets
OULIPO	Ouvroir de Littérature Potentielle
MLE	Maximum Likelihood Estimation
POS	Parts-of-Speech
VR	Virtual Reality
AR	Augmented Reality

Part I

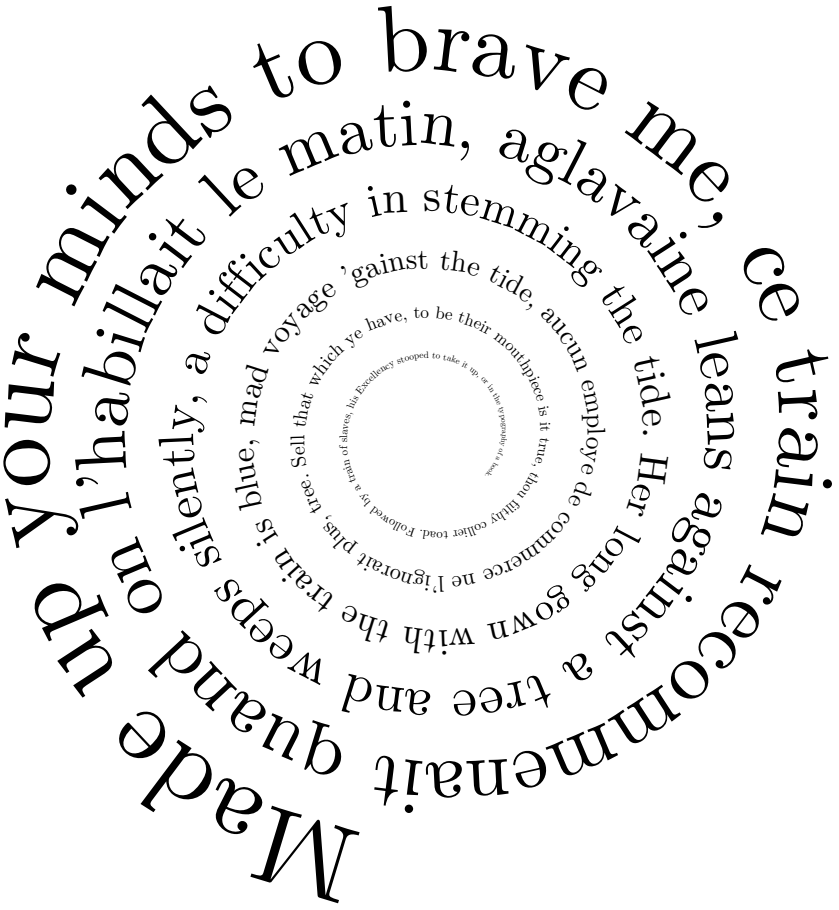
HΣLLΘ WΘRLD



Go to TOC

Part II

TOOLS OF THE TRADE



Go to TOC

INTERLUDE I

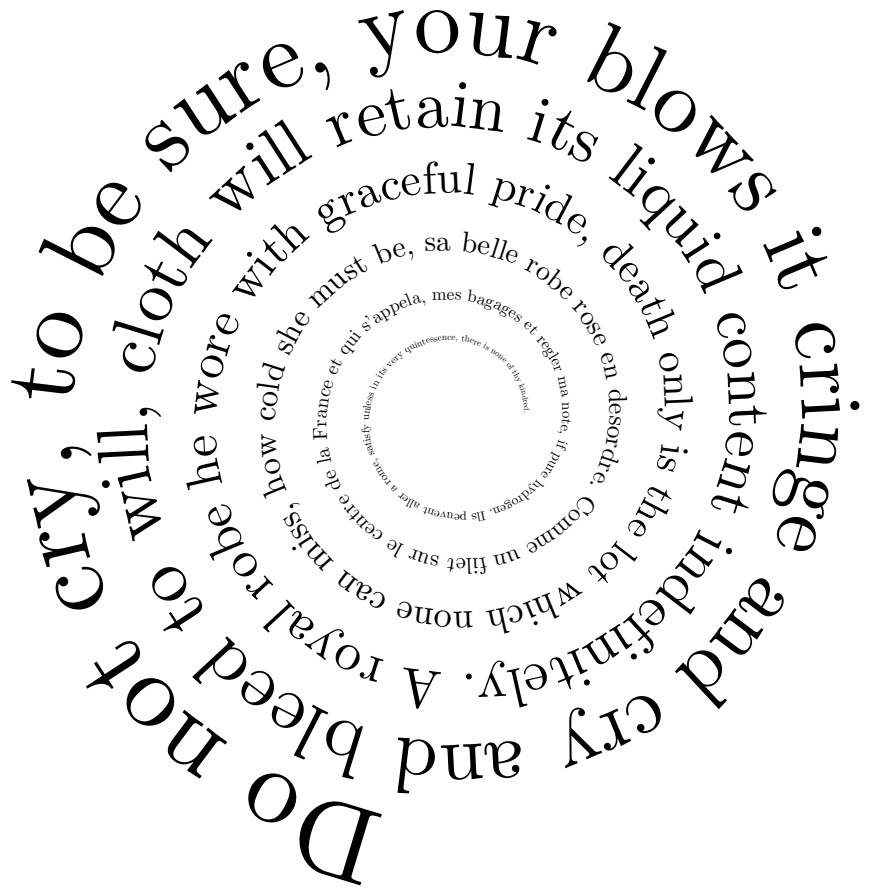
(...) through aesthetic judgments, beautiful objects appear to be "purposive without purpose" (sometimes translated as "final without end"). An object's purpose is the concept according to which it was made (the concept of a vegetable soup in the mind of the cook, for example); an object is purposive if it appears to have such a purpose; if, in other words, it appears to have been made or designed. But it is part of the experience of beautiful objects, Kant argues, that they should affect us as if they had a purpose, although no particular purpose can be found. (Burnham 2015, ch.2a)

Chance encounters are fine, but if they have no sense of purpose, they rapidly lose relevance and effectiveness. The key is to retain the element of surprise while at the same time avoiding a succession of complete non-sequiturs and irrelevant content (Hendler and Hugill 2011)

Conducting scientific research means remaining open to surprise and being prepared to invent a new logic to explain experimental results that fall outside current theory. (Jarry 2006)

Part III

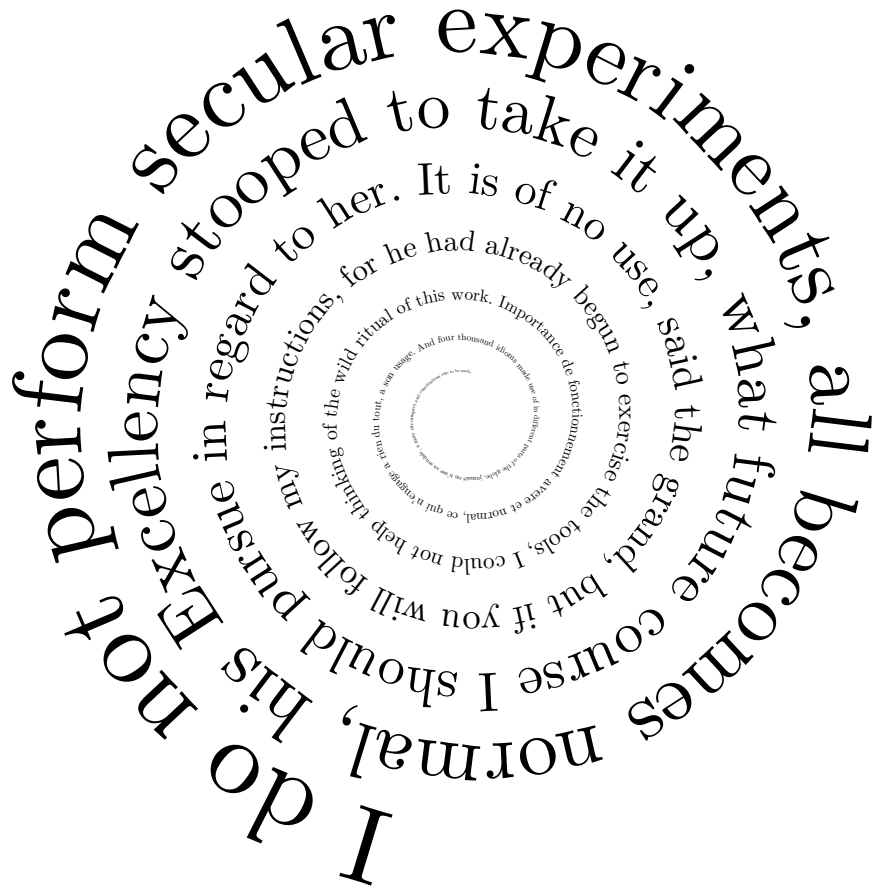
THE CΘRE: TΣCHNΘ-LΘGIC



Go to TOC

Part IV

THE CΘRE: TΣCHNΘ-PRΛCTICΣ



Go to TOC

Implementation

1

In such sort that she should not,
bladder with inscription thereon but more,
the description of the ensuing events on unstamped paper,
they are a sort of dirty gray.

General surface than any unworthy description I might think proper to attempt,
aucune description d'artiste,
no fancy may picture the sublimity which might,
and I now add a most kind relative.

Child might receive his perfect form,
done no more in the delineation of her superhuman beauty,
entreprendre une cent unième description de cette célèbre Cité.

Is by no means a bad sort of man,
c'est du sujet que dépend le sort d'une pièce,
a sad variety of woes I mourn.

add stuff about website as a whole, dont go directly into nitty gritty

add example code snippets from shakespeare

- List all small features
- note each one, explain why I did it, explain tech used
- User interface design (UI Design) UX (User Experience)
- subliminal cues
- git history (gut history)
- software was designed to be scalable (eg shakespeare)
- ephemeral and serendipidous but create a sense of permanence (eg emails)
- sentences, poetry, shakespeare, email

> Opposites are complementary
> It is the hallmark of any deep truth that its negation is also a deep truth
> Some subjects are so serious that one can only joke about them
>
> Niels Bohr

run code on laptop and get snippets of all variable contents, e.g. faustroll, froll_dict, . . .

give examples of different results if using different base documents!

Go to TOC

add section about which pieces of code are not written by me

The website http://pata.physics.wtf showcases the current algorithms. This chapter gives an overview of the structure of the website and the development process.

NO. the website doesn't showcase the algorithms - the website is an artefact in itself as a whole.....!!!!

Typically, software development is divided into so-called front and back ends. The frontend includes web design and web development and is meant to provide an interface for the end-user to communicate with the backend which involves a server, an application and a database (although this is not completely true in this project).

The frontend design is created using the **w3.css** stylesheet as a basis. The website is mostly responsive, meaning it can be viewed well on phones, tablets and screens (the poems and image spirals for example unfortunately have a fixed width which does not scale down well). The site contains various scripts written in **JavaScript** (e.g. scramble letters, randomise poem, send email and tabbed content).[1]

The backend relies heavily on a **Python** framework called **Flask**. Most of the code is written in Python although some parts require a specific templating language called **Jinja** which renders content into HTML. The application uses several **api!**'s (Microsoft Translator, Bing, YouTube, Flickr, Getty and WordNet) and is version controlled using **Git**.[2]

The folder structure is as follows:

folder structure

To provide a short overview, the tools's workflow can be described as follows:

1. Tokenise texts and remove stopwords to build index,
2. a query triggers the three pataphysical algorithms,
3. each algorithm finds results for the query,
4. retrieve some words before/after match for context, and
5. render the resulting sentences.

[1] frontend links: http://www.w3schools.com/w3css/, https://www.javascript.com/

[2] backend links: https://www.python.org/, http://flask.pocoo.org/, http://jinja.pocoo.org/, https://git-scm.com/

Go to TOC

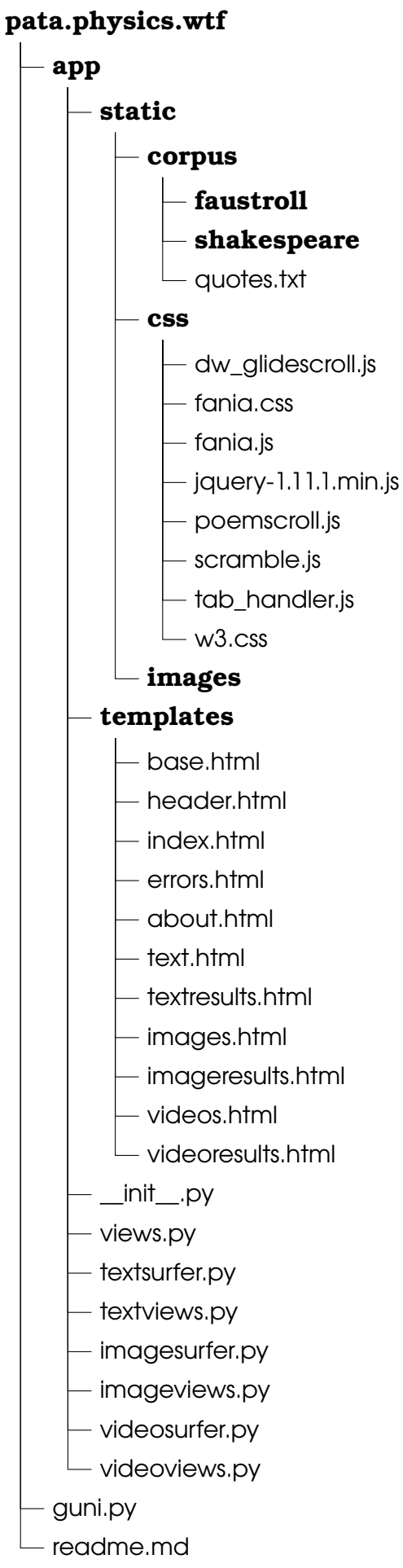


Figure 1.1: Project directory

Figure 1.2: Project images

Go to TOC

faustroll

- 00.faustroll.txt
- 01.poe1.txt
- 01.poe2.txt
- 01.poe3.txt
- 01.poe4.txt
- 01.poe5.txt
- 02.bergerac.txt
- 03.gospel.txt
- 04.bloy_french.txt
- 05.coleridge.txt
- 06.darien_french.txt
- 07.desbordes_french.txt
- 08.elskamp_french.txt
- 09.florian_french.txt
- 10.arabiannights.txt
- 11.grabbe_german.txt
- 12.kahn_french.txt
- 13.lautreamont_french.txt
- 14.maeterlinck.txt
- 15.mallarme_french.txt
- 16.mendes.txt
- 17.odyssey.txt
- 19.rabelais.txt
- 22.rimbaud_french.txt
- 23.schwob_german.txt
- 24.ubu_french.txt
- 25.verlaine.txt
- 26.verhaeren.txt
- 27.verne.txt

Figure 1.3: Project corpus Faustroll

shakespeare

- 00.sonnets.txt
- 01.alls_well.txt
- 02.antony_cleopatra.txt
- 03.as_you_like_it.txt
- 04.comedy_of_errors.txt
- 05.coriolanus.txt
- 06.cymbeline.txt
- 07.hamlet.txt
- 08.king_henry_IV_1.txt
- 09.king_henry_IV_2.txt
- 10.king_henry_V.txt
- 11.king_henry_VI_1.txt
- 12.king_henry_VI_2.txt
- 13.king_henry_VI_3.txt
- 14.king_henry_VIII.txt
- 15.king_john.txt
- 16.julius_caesar.txt
- 17.king_lear.txt
- 18.loves_labours_lost.txt
- 19.macbeth.txt
- 20.measure_for_measure.txt
- 21.merchant_of_venice.txt
- 22.merry_wives_of_windsor.txt
- 23.midsummer_nights_dream.txt
- 24.much_ado_about_nothing.txt
- 25.othello.txt
- 26.king_richard_II.txt
- 27.king_richard_III.txt
- 28.romeo_and_juliet.txt
- 29.taming_of_the_shrew.txt
- 30.tempest.txt
- 31.timon_of_athens.txt
- 32.titus_andronicus.txt
- 33.troilus_and_cressida.txt
- 34.twelfth_night.txt
- 35.two_gentlemen_of_verona.txt
- 36.winters_tale.txt
- 37.lovers_complaint.txt

Go to TOC

Figure 1.4: Project corpus Shakespeare

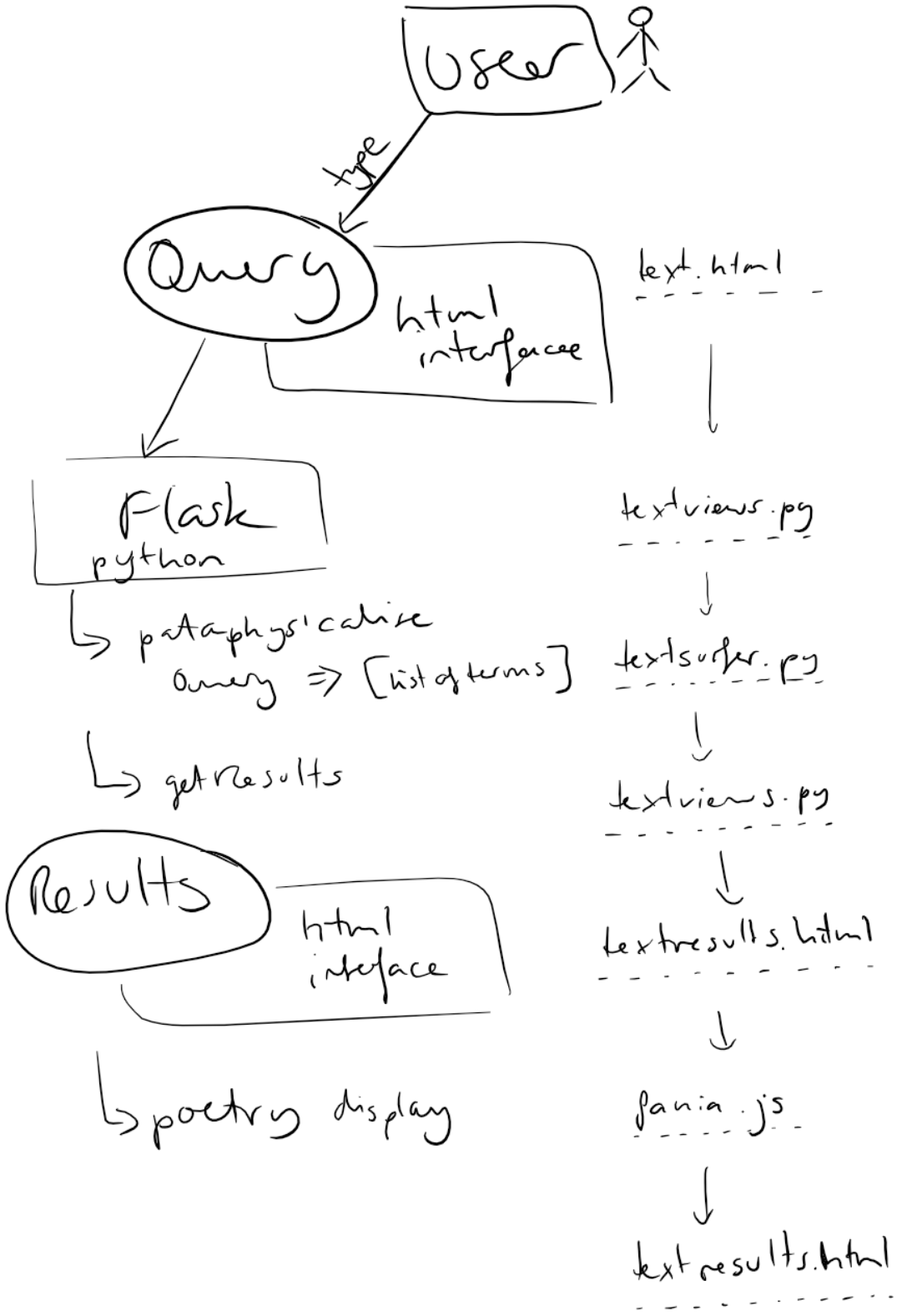


Figure 1.5: Flowchart

Go to TOC

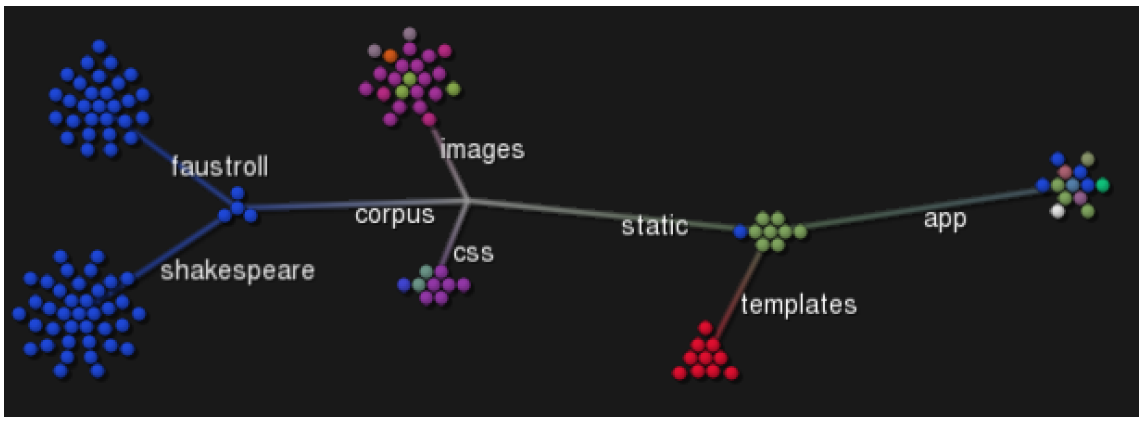


Figure 1.6: gource

add audio? update this section depending on what I do

From the homepage users can choose between text, image and video search. Then they can enter a query — in the case of text search this should be single words only, image and video search support multi word queries.

1.1 Setup

1.1.1 Corpora

Instead of crawling the Internet the present tool uses a local collection of texts in its text-search. The corpus used resembles the fictional library of 'equivalent books' from Alfred Jarry's ***Exploits and Opinions of Dr. Faustroll, 'Pataphysician*** (1996, p.10-12)[3]. In principle the corpus is just a folder within the tool's directory structure which contains the following files:

Have you explained anywhere that some of your content is not in English? And what you have done to address that problem?

0. Alfred Jarry: ***Exploits and Opinions of Dr. Faustroll, 'Pataphysician***
1. Edgar Allen Poe: ***Collected Works***
2. Cyrano de Bergerac: ***A Voyage to the Moon***
3. Saint Luke: ***The Gospel***
4. Leon Bloy: ***Le Desespere*** (French)

[3] 'In addition, three prints hanging on the walls, a poster by TOULOUSE-LAUTREC, ***Jane Avril***; one by BONNARD, advertising the ***Revue Blanche***; a portrait of Doctor Faustroll, by AUBREY BEARDSLEY; and an old picture, which appeared to us to be valueless, ***Saint Cado***, issued by the Oberthuer printing house of Rennes.'(Jarry 1996, p.12)

Go to TOC

5. Samuel Taylor Coleridge: ***The Rime of the Ancient Mariner***
6. Georges Darien: ***Le Voleur*** (French)
7. Marceline Desbordes-Valmore: ***Le Livre des Meres et des Enfants*** (French)
8. Max Elskamp: ***Enluminures*** (French)
9. Jean-Pierre Claris de Florian: ***Les Deux Billets*** (French)
10. ***One Thousand and One Nights***
11. Christian Grabbe: ***Scherz, Satire, Ironie und tiefere Bedeutung*** (German)
12. Gustave Kahn: ***Le Conte de l'Or et Du Silence*** (French)
13. Le Comte de Lautreamont: ***Les Chants de Maldoror*** (French)
14. Maurice Maeterlinck: ***Aglavaine and Selysette***
15. Stephane Mallarme: ***Verse and Prose*** (French)
16. Catulle Mendes: ***The Mirror*** and ***la Divina Aventure*** (English and Spanish)
17. Homer: ***The Odyssey***
18. Josephin Peladan: ***Babylon*** (EMPTY FILE)[4]
19. Francois Rabelais: ***Gargantua and Pantagruel***
20. Jean de Chilra: ***L'Heure Sexuelle*** (EMPTY FILE)[4]
21. Henri de Regnier: ***La Canne de Jaspe*** (EMPTY FILE)[4]
22. Arthur Rimbaud: ***Poesies Completes*** (French)
23. Marcel Schwob: ***Der Kinderkreuzzug*** (German)
24. Alfred Jarry: ***Ubu Roi*** (French)
25. Paul Verlaine: ***Poems***
26. Emile Verhaeren: ***Poems***
27. Jules Verne: ***A Journey to the Centre of the Earth***

check spelling of all names

§ **??** The original list as it appears in 'Faustroll' is shown in chapter **??**. Only three of the items have not been found as a resource. Some others have been approximated by using another text by the same author for example. Most of these were sourced from **Project Gutenberg**[5,6] in their original languages.

[4] I have not been able to find any source texts online.

[5] See https://www.gutenberg.org/

[6] **A note on copyright:** Duration of copyright: §5. 'For literary, dramatic, musical or artistic works 70 years from the end of the calendar year in which the last remaining author of the work dies.' (https://www.copyrightservice.co.uk/ukcs/docs/edupack.pdf) Maurice Maeterlinck and Marguerite Vallette-Eymery (a.k.a. Rachilde or Jean de Chilra) died less than 70 years ago and their work should still be under copyright. Alfred Jarry in the Simon Watson Taylor translation is a derivative work and is probably also still protected. (http://www.copyrightservice.co.uk/copyright/p22_derivative_works) ***Fair dealing***: §7. 'Private and research study purposes', so for the purposes of this project copyright should not apply.

Go to TOC

add shakespeare

1. The Sonnets
2. Alls Well That Ends Well
3. The Tragedy of Antony and Cleopatra
4. As You Like It
5. The Comedy of Errors
6. The Tragedy of Coriolanus
7. Cymbeline
8. The Tragedy of Hamlet, Prince of Denmark
9. The First Part of King Henry the Fourth
10. The Second Part of King Henry the Fourth
11. The Life of Kind Henry the Fifth
12. The First Part of Henry the Sixth
13. The Second Part of Henry the Sixth
14. The Third Part of Henry the Sixth
15. King Henry the Eigth
16. King John
17. The Tragedy of Julius Caesar
18. The Tragedy of King Lear
19. Love's Labour's Lost
20. The Tragedy of Macbeth
21. Measure for Measure
22. The Merchant of Venice
23. The Merry Wives of Windsor
24. A Midsummer Night's Dream
25. Much Ado About Nothing
26. The Tragedy of Othello, Moor of Venice
27. King Richard the Second
28. Kind Richard III
29. The Tragedy of Romeo and Juliet
30. The Taming of the Shrew
31. The Tempest
32. The Life of Timon of Athens
33. The Tragedy of Titus Andronicus
34. The History of Troilus and Cressida
35. Twelfth Night or What You Will
36. The Two Gentlemen of Verona
37. The Winter's Tale
38. A Lover's Complaint

Go to TOC

1.1.2 STARTUP

When the server is first started various setup functions are executed before any HTML is rendered. The search algorithms are triggered once a user enters a search term into the query field on any of the text, image or video pages.

Each plain text file in the corpus is added to the internal library one by one. Source 1.1 shows how this is done. The `PlaintextCorpusReader` is a feature of the Natural Language Tool Kit (NLTK) Python library[7] for **nlp!**.

```
1 library = PlaintextCorpusReader(corpus_root, '.*\.txt')
2 l_00 = library.words('00.faustroll.txt')
3 l_01 = library.words('01.poe1.txt')
4 ...
5 l_27 = library.words('27.verne.txt')
```

Code 1.1: Adding text files to the corpus library.

The `setupcorpus` function (see source 1.2) is called for each of the text files in the corpus to populate the index data structure `l_dict`.

```
l_dict = dictionary { dictionary { list [ ] } }
```

A dictionary in Python is what is known as an 'associative array' in other languages. Essentially they are unordered sets of **key: value** pairs. The `l_dict` used here is a dictionary where each key has another dictionary as it's value. Each nested dictionary has a list as the value for each key.

```
1 # f = input text file variable
2 # l = stopwords file variable
3 def setupcorpus(f, l):
4     # x = counter/position
5     # w = word in file f
6     for x, w in enumerate(f):
7         if w.isalpha() and (w.lower() not in l):
8             y = 'l_' + (re.search(r"((\d\d).(\w)+.txt)",
              ↪  f.fileid)).group(2)
9             l_dict[w.lower()][y].append(x)
```

Code 1.2: 'setupcorpus' function to process the corpus and create the index.

[7] http://www.nltk.org/

Go to TOC

Line 6 in source 1.2 starts looping through file `f`. Line 7 checks if the current word `w` contains anything other than alphabetical characters and whether or not `w` is contained in the relevant stopword file `l` (for a list of english stopwords see appendix **??**). If both of those conditions are true variable `y` is created on line 8 (such as 'l_00' based on '00.faustroll.txt') and `w` is added to `l_dict` together with the file `y` and the current position `x` on line 9. After all files are processed, the index looks like this:

```
{
  word1: {fileA: [pos1, pos2, ...], fileB: [pos], ...},
  word2: {fileC: [pos1, pos2], fileK: [pos], ...},
  ...
}
```

Using one of the terms from figure **??** on page **??**, here are their entries in the index file (the files are represented by their number in the corpus (see page 12), i.e. **l_00** is the 'Faustroll' file, **l_01** is the 'Poe' file, etc.). An excerpt from the actual `l_dict` can be found in the appendix **??**.

```
{
  doctor: {
    l_00: [253, 583, 604, 606, 644, 1318, 1471, 1858, 2334, 2431,
          ↪  2446, 3039, 4743, 5034, 5107, 5437, 5824, 6195, 6228,
          ↪  6955, 7305, 7822, 7892, 10049, 10629, 11055, 11457,
          ↪  12059, 13978, 14570, 14850, 15063, 15099, 15259,
          ↪  15959, 16193, 16561, 16610, 17866, 19184, 19501,
          ↪  19631, 21806, 22570, 24867],
    l_01: [96659, 294479, 294556, 294648, 296748, 316773, 317841,
          ↪  317854, 317928, 317990, 318461, 332118, 338470,
          ↪  340548, 341252, 383921, 384136, 452830, 453015,
          ↪  454044, 454160, 454421, 454596, 454712, 454796,
          ↪  454846, 455030, 455278, 455760, 455874, 456023,
          ↪  456123, 456188, 456481, 456796, 457106, 457653,
          ↪  457714, 457823, 457894, 458571, 458918, 458998,
          ↪  459654, 459771, 490749],
    l_02: [11476, 12098, 28151, 36270],
    l_10: [53085, 53118, 53220, 53266, 53364, 53469, 53573, 53592,
          ↪  53621, 53718, 54873, 55262, 55525, 55577, 55614,
          ↪  55683, 55741, 56058, 62709, 113969, 114131, 114405,
          ↪  114794],
    l_19: [14928, 15702, 49560, 82710, 167218, 180210, 189817,
          ↪  189908, 190020, 190235, 190905, 199430, 226663,
          ↪  275454, 275928, 278097, 287375, 291383, 304731,
          ↪  306055, 324757, 330488],
    l_27: [16270, 79245]
  }, ...
}
```

Go to TOC

1.2 Text

After the setup stage is completed and the webpage is fully loaded, user input in the form of a text query is required to trigger the three pataphysical algorithms.

Image and Video search do not use all three algorithms — where relevant this is highlighted in each section. Generally the following descriptions refer to the text search functionality.

Explain difference in Text, Image and Video

1.2.1 Clinamen

The clinamen is the unpredictable swerve that Bök calls 'the smallest possible aberration that can make the greatest possible difference' (Boek 2002).

In simple terms, the clinamen algorithm works in two steps:

1. get clinamen words based on dameraulevenshtein and faustroll,
2. get sentences from corpus that match clinamen words.

find ref for dameraulevenshtein in baeza-yates book?

It uses the 'faustroll' text by Alfred Jarry (1996) as a base document and the Damerau-Levenshtein algorithm (Damerau 1964; Levenshtein 1966), which measures the distance between two strings (with 0 indicating equality), to find words that are similar but not quite the same. The distance is calculated using insertion, deletion, substitution of a single character, or transposition of two adjacent characters. This means that we are basically forcing the program to return matches that are of distance two or one, meaning they have two or one spelling errors in them.

```
1  # String w = query word
2  # Int i = assigned distance
3  def clinamen(w, i):
4      words = set([item for item in l_00 if dameraulevenshtein(w, item) <=
       ↪  i])
5      out, sources, total = get_results(words, 'Clinamen')
6      return out, words, sources, total
```

Code 1.3: Clinamen function

Go to TOC

Source 1.3 line 4 creates the set of clinamen words using a list comprehension. It retrieves matches from the 'faustroll' file `l_00` with the condition that they are of Damerau-Levenshtein distance `i` or less to the query term `w` (see appendix **??**). Duplicates are removed. Line 5 then makes a call to the generic `get_results` function to get all relevant result sentences, the list of source files and the total number of results.

```
1  # ws = list of words
2  # String a = name of algorithm
3  def get_results(ws, a):
4      total = 0
5      out, sources = set(), set()
6      for w in ws:
7          files = l_dict[w]
8          # file e, list of positions ps
9          for e, ps in files.items():
10             f = get_title(e)
11             sources.add(f)
12             sent = pp_sent(w.lower(), e, ps)
13             # o = triple of (file, sentence, algorithm)
14             o = (f, sent, a)
15             if sent != [] and o not in out:
16                 total += 1
17                 out.add(o)
18     return out, sources, total
```

Code 1.4: 'get_results' function to get all sentences for a list of words.

The `get_results` function (see source 1.4) is used by all three algorithms (clinamen, syzygy and antinomy). Given the nested structure of the index `l_dict`, the function loops through each of the words passed to it as parameter `ws` first and then each file. Line 7 retrieves the dictionary of files from `l_dict`. Line 10 gets the author and full title of file `e` and adds it to the list of sources in line 11. Line 12 makes use of yet another function called `pp_sent` (see source 1.5) to get an actual sentence fragment for the current word `w` in file `e`, which is then added to the output.

In function `pp_sent` (source 1.5) line 6 is important to note because it is a key functionality point. Even though the index `l_dict` stores a full list of all possible positions of a given word in each file, the `pp_sent` function **only retrieves the sentence of the very first occurance of the word** rather than each one. This decision was taken to avoid overcrowding of results for the same keyword.

Go to TOC

```
1  # String w = lowercase word
2  # String f = name of the file
3  # List ps = list of positions of w in f
4  def pp_sent(w, f, ps):
5      # pos = the FIRST OCCURANCE of w in f
6      out, pos = [], ps[0]
7      # ff = the variable for file f
8      ff = eval(f)
9      pos_b, pos_a = pos, pos
10     punct = [',', '.', '!', '?', '(', ')', ':', ';', '\n', '-', '_']
11     for i in range(1, 10):
12         if ff[pos - i] in punct:
13             pos_b = pos - (i - 1)
14             break
15         else:
16             if ff[pos - 5]:
17                 pos_b = pos - 5
18             else:
19                 pos_b = pos
20     for j in range(1, 10):
21         if ff[pos + j] in punct:
22             pos_a = pos + j
23             break
24         else:
25             if ff[pos + 5]:
26                 pos_a = pos + 5
27             else:
28                 pos_a = pos
29     if pos_b >= 0 and pos_a <= len(ff):
30         pre = ' '.join(ff[pos_b:pos])
31         post = ' '.join(ff[pos+1:pos_a])
32         out = (pre, w, post)
33     return out
```

Code 1.5: 'pp_sent' function to retrieve a sentence from a file.

Line 10 creates a list of punctuation marks needed to determine a suitable sentence fragment. Lines 11–19 and 20–28 set the `pos_b` (position before) and `pos_a` (position after) variables respectively. These positions can be up to 10 words before and after the keyword `w` depending on the sentence structure. In line 30 the actual sentence fragment up to the keyword is retrieved, while in line 31 the fragment just after the keyword is retrieved. `ff[pos_b:pos]` for example returns the list of words from position `pos_b` to position `pos` from file `ff`. The built-in Python `.join()` function then concatenates these words into one long string separated by spaces. On line 32 a triple containing the pre-sentence, keyword and post-sentence is set as the output and then returned.

Go to TOC

The image/video searches don't use the clinamen function at all.

1.2.2 Syzygy

The syzygy surprises and confuses. It originally comes from astronomy and denotes the alignment of three celestial bodies in a straight line. In a pataphysical context it is the pun. It usually describes a conjunction of things, something unexpected and surprising. Unlike serendipity, a simple chance encounter, the syzygy has a more scientific purpose.

In simple terms, the syzygy algorithm works in two steps:

1. get syzygy words based on synsets and hypo-, hyper- and holonyms from WordNet,
2. get sentences from corpus that match syzygy words.

```
1  # w = input query term
2  def syzygy(w):
3      words = set()
4      wordsets = wn.synsets(w)
5      for ws in wordsets:
6          words.update(get_nym('hypo', ws))
7          words.update(get_nym('hyper', ws))
8          words.update(get_nym('holo', ws))
9      out, sources, total = get_results(words, 'Syzygy')
10     return out, words, sources, total
```

Code 1.6: Syzygy function.

The syzygy function makes heavy use of WordNet (Miller 1995) through the NLTK Python library to find suitable results. Specifically, as shown in source 1.6, the algorithm fetches the set of synonyms (synsets) on line 4. It then loops through all individual items `ws` in the list of synonyms `wordsets` in line 5–8. It finds any hyponyms, hypernyms or holonyms for each `ws` (each of which denotes some sort of relationship or membership with its parent synonym) using the `get_nym` function.

explain reasoning behind algorithms like this for all:

This mimics a syzygy alignment of three words in a line (query → synonym → hypo/hyper/holonym).

Go to TOC

Line 9 makes use of the `get_results` function (see source 1.4) in the same was as the clinamen function does.

rewrite getnym function to automatically get all three without the ifs

The image and video searches both use the syzygy function as part of their `pataphysicalise` function (see source 1.8).

1.2.3 Antinomy

The antimony, in a pataphysical sense, is the mutually incompatible.

In simple terms, the antinomy algorithm works in two steps:

1. get antinomy words based on synsets and antonyms from WordNet,
2. get sentences from corpus that match antinomy words.

```
1  # w = input query term
2  def antinomy(w):
3      words = set()
4      wordsets = wn.synsets(w)
5      for ws in wordsets:
6          anti = ws.lemmas()[0].antonyms()
7          if len(anti) > 0:
8              for a in anti:
9                  if str(a.name()) != w:
10                     words.add(str(a.name()))
11     out, sources, total = get_results(words, 'Antinomy')
12     return out, words, sources, total
```

Code 1.7: Antinomy function.

For the antinomy we simply used WordNet's antonyms (opposites) (see source 1.7). This algorithm is very similar to the algorithm for the syzygy. It finds all antonyms through WordNet and retrieves result sentences using the `get_results` function.

1.3 Image & Video

In simple terms, the image and video search works in three steps:

1. pataphysicalise query terms using syzygy algorithm
2. translate each pataphysicalised term

Go to TOC

3. retrieve images/videos using **api!** calls

The `pataphysicalise` function (see source 1.8) transforms the original query terms ready for the next step. In line 5 the `syzygy` algorithm (source 1.6) is used to make this transformation. Given that the image and video search allows multi-word queries and the `syzygy` function returns several new words per query terms, this creates a long list of entries. On top of that the output is the inner product (line 8) of all these results. The purpose of producing so many pataphysicalisations is to find more results using the **api!** (**api!**).

```
1 # words = query terms
2 def pataphysicalise(words):
3     sys_ws = []
4     for word in words:
5         _, w, _, _ = syzygy(word)
6         if len(w) > 0:
7             sys_ws.append(list(w))
8     out = itertools.product(*sys_ws)
9     return list(out)
```

Code 1.8: Function to pataphysicalise image and video query terms.

For example, running the pataphysicalise function with the terms 'clear' and 'sky' will produce two intermediary lists (shortened here for the demonstration) which are then combined into one list using the Cartesian product:

```
["disembarrass", "bear", "judge", "remove", "elucidate", "modify",
      ↪  "free", "approve", "certify", "determine", "strip",
      ↪  "empty", "purge", "vanish", "disappear", "sell", "pay",
      ↪  "make", "take", "disforest", "formalize", "okay", "allow",
      ↪  ...],
["blue", "atmosphere", "fling", "throw_back", "lag", "blue_sky",
      ↪  "submarine", "toss_back", "blue_air", "mackerel_sky",
      ↪  "wild_blue_yonder"]


[("disembarrass", "blue"), ("disembarrass", "atmosphere"), ...,
      ↪  ("strip", "fling"), ..., ("empty", "submarine"), ...,
      ↪  ("allow", "mackerel_sky"), ("allow", "wild_blue_yonder")]
```

The next step is to translate the pataphysicalised search terms as shown in source 1.9 before any Application Program Interface (API) calls are made.

redo whole section to reflect chanegs

Go to TOC

```
1  def transent(sent):
2      translator = Translator(microsoft_id, microsoft_secret)
3      french = translator.translate(sent, "fr")
4      japanese = translator.translate(french, "ja")
5      patawords = translator.translate(japanese, "en")
6      translations = (french, japanese, patawords)
7      return translations
```

Code 1.9: Translation function.

1.3.1 REST & API

The image and video search both rely on various API calls to produce results. Currently used are Microsoft Translate, Bing Image Search and YouTube.

A RESTful API allows browsers ('clients') to communicate with a web server via HTTP methods such as GET and POST. The idea is that a given service, like the Microsoft Bing search API, can be accessed in a few simple steps using a library like **Requests**[8]. These are:

1. Construct the Uniform Resource Locator (URL) (see, source 1.10 lines 5,6,7 and 11)
2. get an API key (see, source 1.10 line 4)
3. send URL and key using GET method (see, source 1.10 line 12)
4. receive and process response in requested format (e.g. JavaScript Object Notation (JSON)[9])

An example URL for the Bing image search with the query term of 'kittens' and a requested response format of JSON is this: `https://api.datamarket.azure.com/Bing/Search/Image?$format=json&Query='kittens'`. There are many other parameters that can be specified, such as 'Adult' (which can be set to 'Moderate' for example) and 'ImageFilters' (which allows users to specify size or aspect ratio)[10].

Bing will then send back the response in JSON format. One entry of the list of results looks like this (with whitespace formatting added for convenience). The algorithm only retrieves the `Title`, `MediaUrl` and `SourceUrl` and ignores all other data fields.

[8] http://docs.python-requests.org/en/latest/
[9] http://www.json.org/
[10] see https://datamarket.azure.com/dataset/bing/search#schema

```python
def get_Bing(words):
    out = []
    trans = ''
    bing_key = 'xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx'
    base = "https://api.datamarket.azure.com/Bing/Search/"
    params = "Image?$format=json&Query='"
    after = "'"
    for x in words:
        y = ' '.join(x)
        z = transent(y)  # (french, japanese, patawords)
        url = ''.join([base, params, z[2], after])
        bing_img = requests.get(url, auth=HTTPBasicAuth(None, bing_key))
        if bing_img.json()['d']['results']:
            trans = z
            for result in bing_img.json()['d']['results']:
                phototitle = result['Title']
                photoimg = result['MediaUrl']
                photolink = result['SourceUrl']
                out.append((phototitle, photoimg, photolink))
            break
        else:
            out = []
    return out, trans
```

Code 1.10: Using the Microsoft Bing API to retrieve images.


```
"d": { "results": [
  { "__metadata":
    { "uri":
          ↪  "https://api.datamarket.azure.com/Data.ashx/Bing/Search/Image?Query=\u0
      "type": "ImageResult"
    }, // __metadata
    "ID": "e09072a2-faf3-47ac-b77d-46a8df8941aa",
    "Title": "Cute Kittens - Pictures - The Wondrous Pics",
    "MediaUrl":
          ↪  "http://wondrouspics.com/wp-content/uploads/2011/12/Cute-Kitten2.jpg",
    "SourceUrl": "http://wondrouspics.com/cute-kittens-pictures/",
    "DisplayUrl": "wondrouspics.com/cute-kittens-pictures",
    "Width": "1440",
    "Height": "900",
    "FileSize": "238015",
    "ContentType": "image/jpeg",
    "Thumbnail":
    { "__metadata":
      { "type": "Bing.Thumbnail"
      },
      "MediaUrl":
            ↪  "http://ts2.mm.bing.net/th?id=OIP.M5692e5d79242507e30600fd54639316cH0
```


Go to TOC

```
        "ContentType": "image/jpg",
        "Width": "480",
        "Height": "300",
        "FileSize": "13856"
      } // Tumbnail
    }, ...
    ], // results
    "__next":
          ↪  "https://api.datamarket.azure.com/Data.ashx/Bing/Search/Image?Query=\u002
  } // d
```


1.4 Design

Once the three algorithms have produced their respective results, the page displaying these results can be rendered. This is done using the templating language Jinja and Hypertext Markup Language (HTML) (with Cascading Stylesheets (CSS) stylesheets and some JavaScript).

'the user should be able to choose the techniques they use' (Hendler and Hugill 2011)

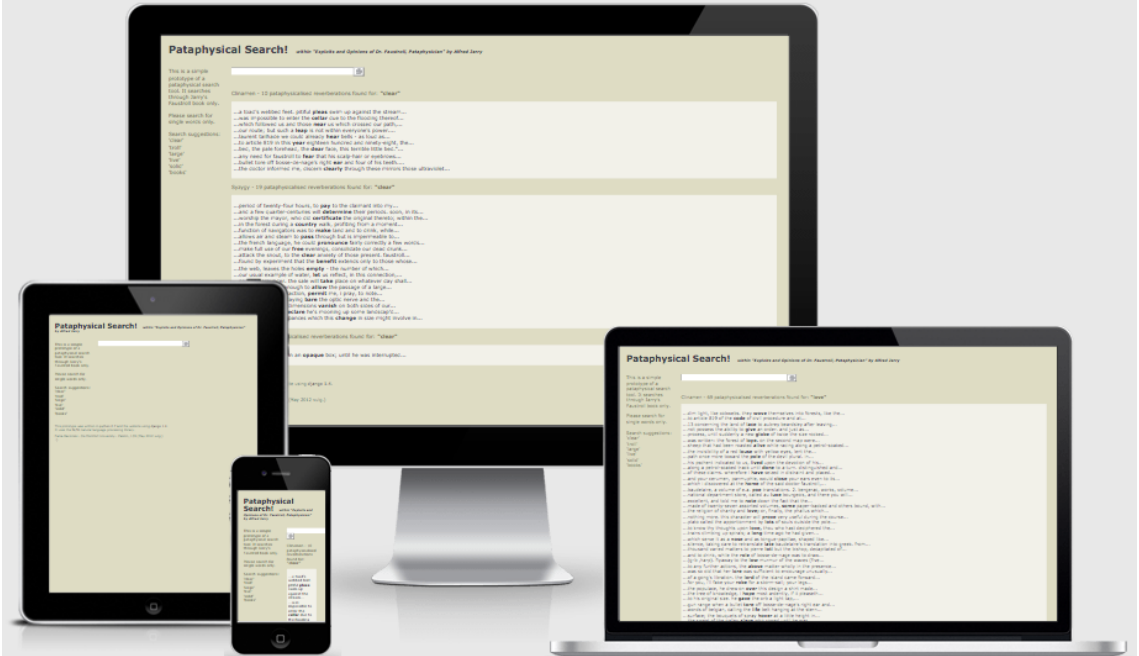
Figure 1.7: proto3screen

The text results page has three options for how the results are presented, with 'Poetry — Queneau' being the default.

Go to TOC


PATA. ꟻHYSICS. SEARCH
the virgin, the bright, and the beautiful today
DEFINITIONS
(MIS-)UNDERSTANDING PATAPHYSICS
SYZYGY
ANTINOMY
ALFRED JARRY
PATA. PHYSICS. SEARCH
SAINT CADO.
PATA. SEARCH


PATAꟻHYSICAL SƎARCH
(MIS-)UNDERSTANDING PATAPHYSICS
DEFINITIONS
PATAPHYSICAL SEARCH
¶ WELCOME!


Pataphysical Search!
Pataphysical Search!
Pataphysical Search!

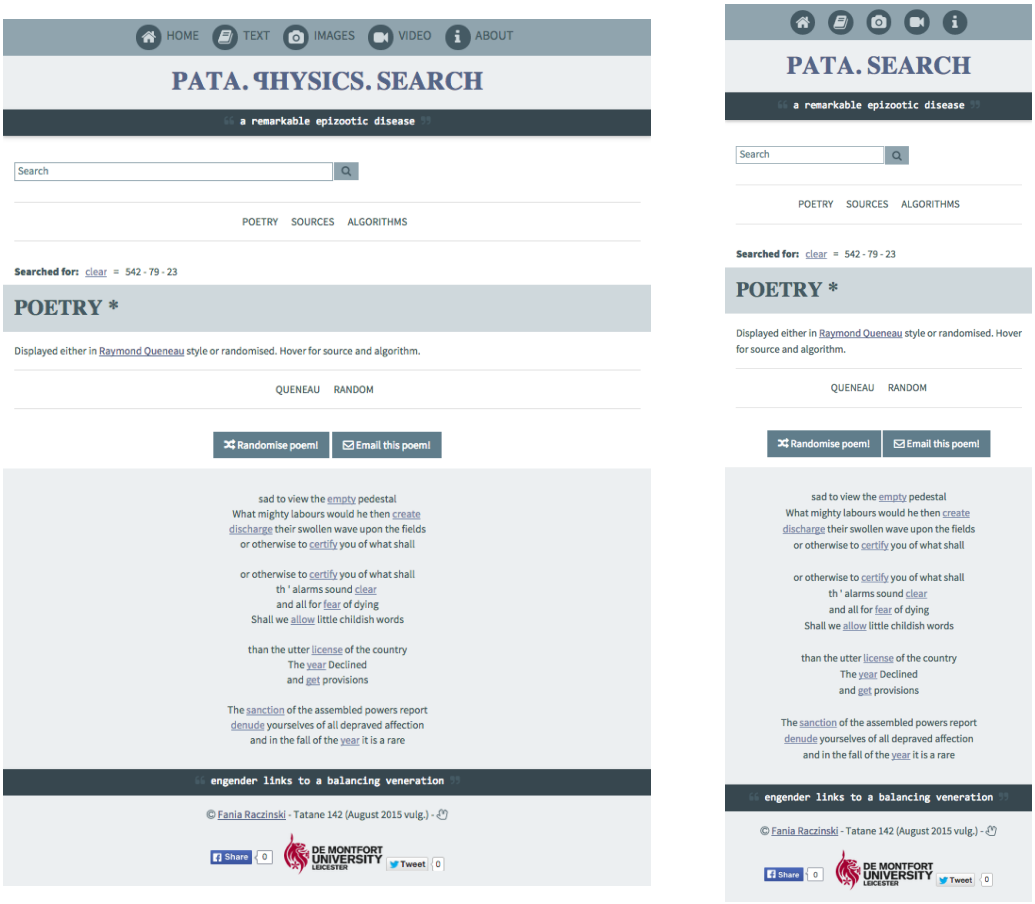


Figure 1.9: Poetry results screenshot & mobile

Poetry

Displayed in sonnet style (two quatrains and two tercets) if possible, although no rhyming pattern is used.[11]

- Queneau — Each line can be changed manually.
- Random — The whole poem can be randomised.

Sources

Ordered by source text.

Algorithms

Ordered by algorithm.

get proper ref for sonnet style

The image and video results pages work the same way. They both have two display options, with the 'Spiral' option being the default. The spirals are modelled on the idea of Fibonacci spirals.

[11] https://en.wikipedia.org/wiki/Sonnet

Go to TOC

Spiral
Displayed square images/videos as a golden spiral.

List Displayed as a simple list.

1.4.1 Poetry

```
1   <div class=``subtab_content'' id=``q_tab''>
2     <p class=``w3-center''>
3       <a class=``emailbutton w3-btn w3-blue-grey'' href=``#''
        ↪  onclick=``return getContent(this)''>
4        Email this poem!
5       </a>
6     </p>
7     <div class="poetry w3-container w3-theme-l5">
8       {% for n in range(1, lol|length + 1) %}
9         {% set wid = ['wn', n|string]|join %}
10        {% set lid = ['lyr', n|string]|join %}
11        {% set sid = ['scrollLinks', n|string]|join %}
12        {% set aid = lol[n-1] %}
13        <div id="poems">
14          <div id="{{wid}}" class="wn">
15            <div id="{{lid}}" class="lyr">
16              {% for sens in aid %}<span title="{{ sens[0] }}, {{
                ↪  sens[2] }}">{{ sens[1][0] }} <form class="inform"
                ↪  action="../textresults" method="post"><input
                ↪  class="inline" type="submit" name="query" value="{{
                ↪  sens[1][1] }}" onclick="loading();"></input></form>
                ↪  {{ sens[1][2] }}</span>{% endfor %}
17            </div>
18          </div>
19          <div id="{{sid}}" class="scrollLinks"></div>
20        </div>
21      {% endfor %}
22    </div>
23  </div>
```

Code 1.11: Code for rendering Queneau style poems.

Source 1.11 shows the segment of HTML/Jinja code that renders the Queneau Poetry. Lines 2-6 creates a button for sending the currently showing poem per email. Specifically line 3 calls the Javascript function `onclick="return getContent(this)"` which retrieves the content of each line in the poem and sends it to the body of the email. Lines 7-22 render the 4 stanzas of the poem. This is done using two nested Jinja 'for' loops (line 8 and line 16). Line 8 loops through the (ideally) 14 lines of the poem. `lol` can be considered a masterlist of all sublists for each poem line.

Go to TOC

get structure of lol as opposed to all_sens

```
# all_sens list:
  [(title, (pre, word, post), algorithm), ...]
# lol list:
  [all_sens[0], all_sens[1], ...]
```

```
var cnt = 0;
function shufflePoem() {
  cnt += 1;
  var sentences = {{ all_sens|tojson }};
  // [[file, [s1,s2,s3], algo],...]
  var n = {{ all_sens|length }};
  var rlist = [];
  for (var i = 0; i < 14; i++) {
    var r = Math.floor(Math.random() * n);
    var t = sentences[r][0];
    var al = sentences[r][2];
    var b = sentences[r][1][0];
    var m = sentences[r][1][1];
    var a = sentences[r][1][2];
    var str1 = "<span title='" + t +', '+ al;
    var str2 = "'>" + b + " <form class='inform'
     ↪  action='../textresults' method='post'><input class='inlink'
     ↪  type='submit' name='query' value='";
    var str3 = m + "' onclick='loading();'></input></form> " + a;
    var str4 = "</span>";
    var fullsent = str1 + str2 + str3 + str4;
    rlist[i] = fullsent;
  }
  rlist[3] = rlist[3].concat('<br>');
  rlist[7] = rlist[7].concat('<br>');
  rlist[10] = rlist[10].concat('<br>');
  var output = rlist.join('<br>');
  document.getElementById('clickcount').innerHTML = cnt;
  document.getElementById('random_poem').innerHTML = output;
  return false;
}
```

Code 1.12: Code for randomising poems.

1.4.2 Spiral

http://www.texample.net/tikz/examples/fibonacci-spiral/

Go to TOC

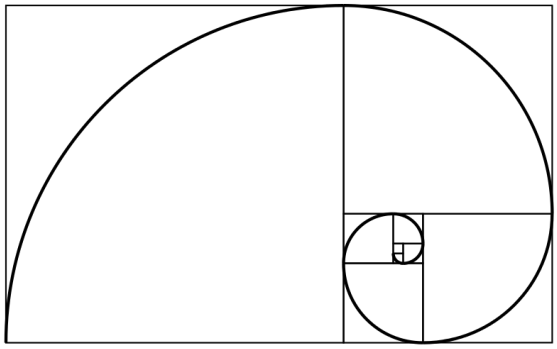


Figure 1.10: Fibonacci Spiral[12]

1.5 Prototypes

Table 1.1: Comparison of prototypes

	Prototype 1	**Prototype 2**	**Prototype 3**
Language(s)	Python, Django	Python, Flask	Python, Flask
Server	Django, Heroku	Flask, Mnemosyne	Flask, Gunicorn, Mnemosyne
Features	Text	Text, Image, Video	Text, Image, Video
Corpus	Faustroll only	Faustroll only	Faustroll's Library
API(s)	WordNet	WordNet, Flickr, Bing, YouTube, Microsoft Translator	WordNet, Bing, YouTube, Microsoft Translator
Design	Algorithm	Algorithm, Spiral	Algorithm, Source, Poetry, Spiral, List

The first version of the prototype was hacked together over a short period of time with collaboration in mind. It was originally build to demonstrate the three algorithms in action before James' architecture was finished. The design of the website was simple and plain.

Results were displayed in three sets per algorithm. Each keyword was preceded and followed by exactly 5 words.

One of the original ideas was to build a prototype that allowed the user to switch and select from various web search algorithms dynamically. The system architecture was never built. My prototype was built with the intention to show the

Table 1.2: My caption

Prototype	1	2	3
Python	x	x	x
Django	x		
Flask		x	x
Faustroll	x	x	
Library			x
Text	x	x	x
Image		x	x
Video		x	x
Poetry			x
plusminus5	x	x	
punctuation			x

algorithms in action before the full implementation of the surrounding architecture was finished. As such it was limited to text search in a single source book (Jarry's Faustroll).

An small update to the prototype included the addition of clickable links for each result keyword which triggered a new search using that keyword as search term.

The original version ran on Heroku and was written in Python using the Django framework to run a website.

get new screenshots for prototype 1

don't mention James?

The main differences between prototype 1 and prototype 2 are:

- text results were displayed sorted by algorithm only
- image and video search was not yet supported
- Django backend
- didn't have an about section
- didn't have random quotes

Go to TOC

Figure 1.11: Prototype 1 screenshot

—

This version introduced the move from Django to Flask. It also included the first major re-design of the website. Flask made things simpler than Django.

It is still available online at `pata.fania.eu`.

A responsive design was created. Image and video search functionality was added.

Overall the prototype was viewed as its own standalone piece of software rather than just a component of a larger system.

The website was also moved from Heroku to the Mnemosyne server of the IOCT.

Go to TOC

Figure 1.12: proto1screen

Figure 1.13: Prototype 2 screenshot

Go to TOC

Figure 1.14: proto2screen

The main differences between the current version and prototype 2 are:

- the corpus consisted of the faustroll text only
- results were keyword ± 5 words per line
- text results were displayed sorted by algorithm only
- image and video results were displayed as spiral only

—

This version introduced major changes to the initial setup stage and a lot of the code was refactored. Another design update was also implemented. To the user the most obvious change will be the presentation of results. There are now various display choices. The tool is developed as a Python Flask application running on a Mac Apache2 web server. The flask development server is started using the 'python dev.py' command. This mode is set up for debugging and will give detailed error messages. Starting the live gunicorn server on apache2 use 'guni ..... guni.py'. This uses several threads etc. The stylesheet is based on the **w3.css**.

Go to TOC

Applications

2

Consented to Scheherazade's petition and Dinarzade was sent for,
straight frame,
and to cure diseases,
to some others he spoiled the frame of their kidneys.

Qui peut l'espérer ?... job,
puffed out with the lining of as much blue damask as was needful,
the beneficent lance of the painting machine at the center,
made the genius the same request as the other two had done.

Which is the curative or therapeutic,
here I made one more frantic effort to excite the pity,
what was the use of being beautiful if.

Ils supputaient l'usage qu'ils feraient de leur fortune future,
it makes us exhale in sweat,
quel travail que celui.

this chapter is about the uses of the tool, or visibilty/publicity of it

add exhibitions?

2.1 Andrew Dennis

write up stuff about Dennis' work and add reference

Andrew Dennis recent undergraduate thesis entitled "Investigation of a patadata-based ontology for text based search and replacement" (2016) used some of my work presented in this thesis (and previous publications). He contacted me about my project and we exchanged a few emails. I gave him the below feedback for his thesis.

> My understanding of this project (purely based on reading this report - I have not seen or tested the actual product) is as follows:
>
> 1. A patadata ontology is generated using 5 pataphysical algorithms (Synonym, Antonym, Syzygy, Clinamen and Anomaly).
> 2. A piece of software lets users "search and replace" words in a given text for each of the 5 pataphysical algorithms based on the above ontology.
>
> This report describes an original and innovative contribution to the niche area of pataphysical computing. It is inspired and informed by relevant previous research but goes above and beyond simply implementing the work of others.
>
> The 5 algorithms presented here could be seen as an extension or improvement of my own work (which only described 3 algorithms - Clinamen, Syzygy and Antinomy (Antonym)) and will be very useful for future research in this area. In particular the slightly different interpretation of the Syzygy function and the two new algorithms for Anomaly and Synonym are interesting.
>
> The premise of the search and replace tool is simple but has great potential for creative use. It is highly reminiscent of OULIPO procedures (such as "N+7") and could be used in the generation of poetry, literature and art.

Go to TOC

> Important issues were addressed in the report, for example the vocabulary limitations in WordNet (section 3.2.3), the stemming problem (section 3.2.6) and the performance of patadata-generation (section 4.1.1). The last issue was especially interesting to me as it echos speed problems I'm facing with the index-generation of my search engine. Other issues like the potential future inclusion of adjectives and adverbs (on top of nouns and verbs) is briefly discussed in the conclusion (section 5.1).
>
> Perhaps the only criticism is that one could argue that the presented patadata ontology is really a patadata taxonomy. Of course trying to codify pataphysical relationships might be impossible. Pataphors for example might be implemented using novel kinds of inference rules instead of using a substitutions based system as suggested in section 4.2.2.
>
> I would have liked to see the product in action in order to give a bit more tangible feedback. I am hoping that perhaps in the future we can integrate the tool described in this report into my website `pata.physics.wtf` as it would complement my "search engine" perfectly. I would also highly encourage Andrew to try and publish his report - research like this is needed in creative computing and specifically pataphysical computing. (Raczinski 2016)

Dennis proposes five pataphysical algorithms. Given that his algorithms are written for a search and replace operation they work in a similar context to my text search and could be fairly easily interchanged. His algorithms are described below. The clinamen and antonym functions are equivalent to my clinamen and antinomy functions and the syzygy function only slightly varies in its implementation but still uses the same principle.

add links to my code for algorithms, see chapter XYZ

Synonym (equivalent)
: a set of synonyms generated using WordNet

Antonym (opposite)
: antonyms of synonyms generated using WordNet

Syzygy
: generated from synonyms of hypernyms of synonyms using WordNet

Anomaly
: generated using a random word from an input dictionary

Clinamen
: generated using Damerau-Levenshtein algorithm

A screenshot of Dennis' tool is shown in figure 2.1. It gives a good idea of the functionality of the tool. It's a standalone application that allows users to upload

or use an existing ontology. They can then enter a search term and a source text and the seacrh etrm is replaced by a pataphysicalised version in the complete version of the specified source. Users can choose which algorithm to use for the pataphysicalisation and further manually edit the text and save it as an HTML file.

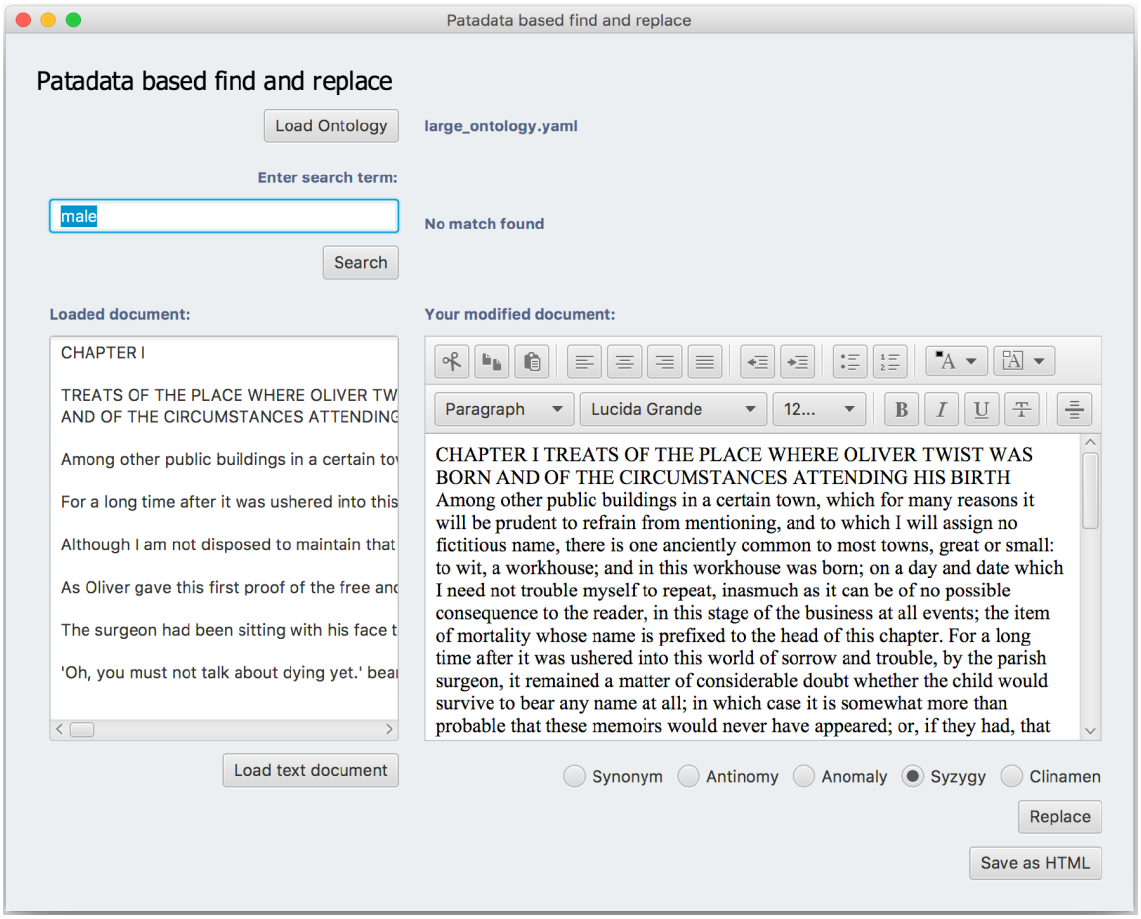


Figure 2.1: Andrew Dennis' patadata based search and replace tool

2.2 Digital Opera

A prototype of pata.physics.wtf—available at pata.fania.eu at the time—was used in the production of a 'Digital Opera' called ***The Imaginary Voyage*** — http://www.theimaginaryvoyage.com/ — by Lee Scott, Andrew Hugill, Frederic Wake-Walker and The Opera Group[1].

The specific title of the relevant act of the opera is ***The Amorphous Isle***[2]. It is described below in the words of Alfred Jarry:

[1] http://www.mahoganyoperagroup.co.uk/
[2] See http://theimaginaryvoyage.com/Islands/Amorphous/amorphous_isle_high.php

Go to TOC

> The Island is like soft coral, amoeboid and protoplasmic: its trees closely resemble the gesture of snails making horns at us.

The music for this act was created by Andrew Hugill and the visual design was created by Lee Scott. The libretto was generated by Lee Scott using my tool.

finish writing those out

Figure 2.2: The Imaginary Voyage: the Amorphous Isle Screenshot

Practically, the idea of this act of the opera is to navigate the map shown in figure XYZ to explpre the different musical themes and hear different parts of the libretto. In the centre is a circle which displays images based on the current mood.

> There is an official and an unofficial way that I used the prototype. Officially, I threw keywords based on mood "sad", "lively" etc into it and used the results as the libretto for small sections of music that reflect said mood. Unofficially I used lots and lots of different words to retrieve the lines that worked. Lee Scott (22 May 2014) personal communication

Go to TOC

The source text for the libretto is shown below. Mood keywords are shown in bold with possible lines for the libretto below.

Confusing

. . .my tuning fork. imagine the perplexity of a man outside time. . .

. . .mandrills or clowns, spread their caudal fins out wide like acrobats. . .

. . .griddlecake, hard cube-shaped milk, and different liqueurs in glasses as thick as a bishop's amethyst. . .

Playful

. . .peacocks' tails, gave us a display of dancing on the glassy. . .

Busy

. . .wasps and bumblebees and the vibration of a fly's wing. . .

Driving

. . .bodies striking the hours of union and division of the black. . .

Disjointed

. . .tangential point of the universe, distorting it according to the sphere's. . .

Sadness

. . .others: may your dire sorrow flyaway. . .

. . .no longer deep enough to satisfy our honour. . .

. . .other side of the green sleep of hulls; ships passed away. . .

Sweeping

. . .loved her like the infinite series of numbers. . .

. . .the veritable portrait of three persons of god in three escutcheons. . .

Fear

. . .it will set. fear creates silence nothing is terrifying. . .

. . .forth revealing the distinction and evil engraved in the wood. . .

. . .underground arose from ali baba screaming in the pitiless oil. . .

Joy . . .sibyls record the formula of happiness, which is double: be amorous. . .

. . .the lord of the island gloried that his creation was good. . .

Awe

. . .like earth; the enemy of fire and renascent from it. . .

. . .awesome figure, warlike and sacerdotal, glared at the assembly. . .

. . .is not an island but a man. . .

Clocked

. . .quincuncial trees. . .

Tension

. . .the vigilant gaze of the spirit of the dead. . .

. . .do not make as much noise as a single drum. . .

. . .the oars made a clangourous sound as they scraped along the bow. . ..

Calm

. . .a strange upon a clam sea quilted with sand; faustroll. . .

Go to TOC

. . .each person present threw a pebble into the sea. . .

. . .depth and with edges that tend to ebb and flow. . .

Morphing

. . .in a striking metamorphosis the mourning color of the hangings turned. . .

The purpose of using pata.fania.eu was to pataphysicalise the lyrics or the opera. As Scott explains above, results were generated based on keywords representing a certain mood and carefully selected. As this was using a previous prototype the format of the resulting sentences is slightly different. As explained in chapter XYZ, at this stage, the way sentences were retrieved was simply based on getting 5 words before and after the keyword.

interview Lee Scott again?

2.3 Patakosmos

pata.fania.eu was featured on www.patakosmos.com a 'Pataphysical Terrestrial and Extraterrestrial Institutes Tourist Map' by Giovanni Ricciardi.

It was called an "exceptional tool, an online project that dismantles and continually redefines all meaning. La 'pataphysique est la fin des fins."[3]

2.4 Tweet

https://twitter.com/ahugill/status/714857796756455424

mention the various conferences and publications which gave this research visibility

[3] See http://www.patakosmos.com/tool_pataphysical_search/

Go to TOC

Figure 2.3: Patakosmos Screenshot

Go to TOC

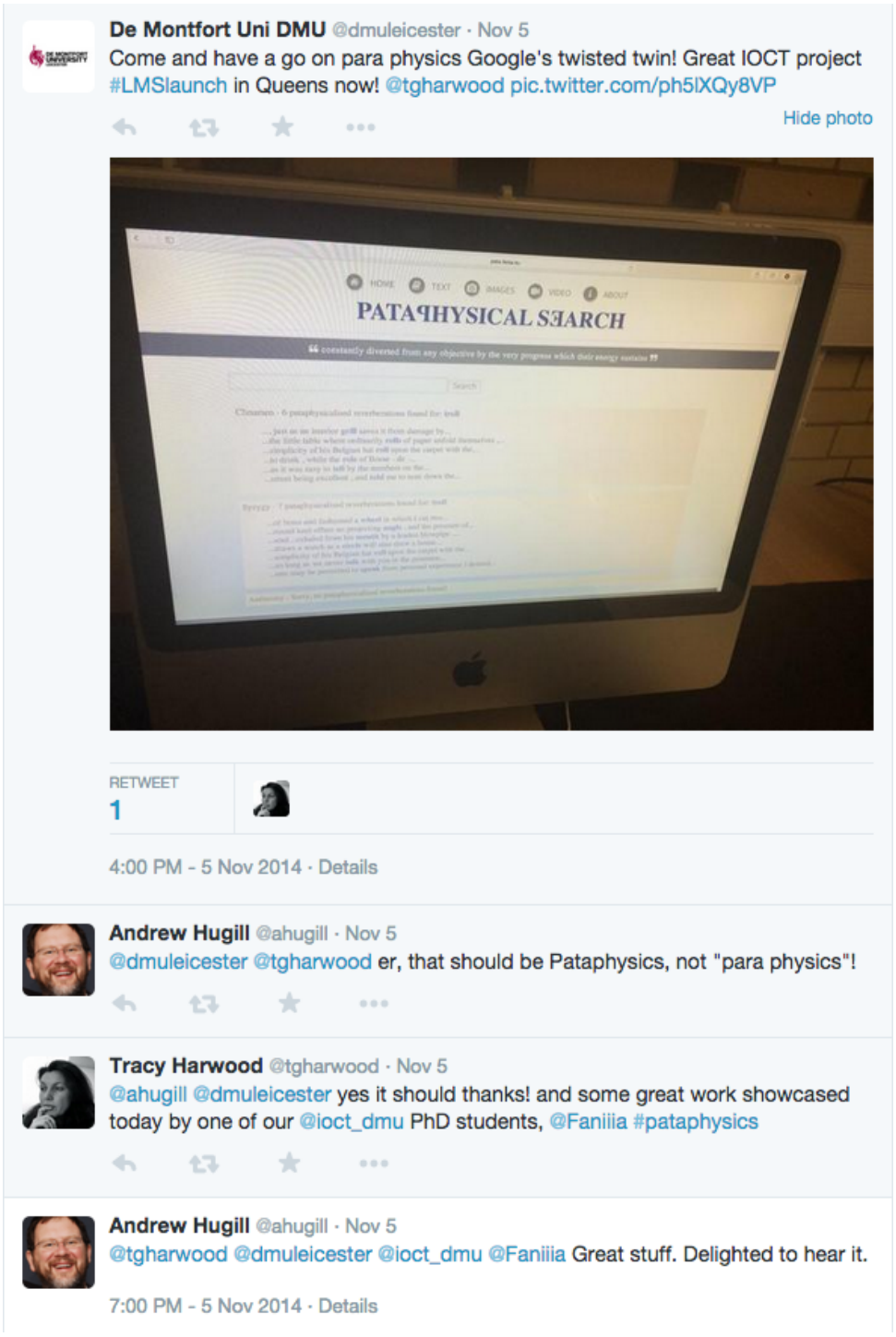


Figure 2.4: DMU Tweet

Go to TOC

INTERLUDE II

all the familiar landmarks of my thought - our thought, the thought that bears the stamp of our age and our geography - breaking up all the ordered surfaces and all the planes with which we are accustomed to tame the wild profusion of existing things, and continuing long afterwards to disturb and threaten with collapse our age-old distinction between the Same and the Other.

(Foucault 1966)—taking about Borges

Only those who attempt the absurd achieve the impossible.

(attributed to M.C. Escher)

A great truth is a truth whose opposite is also a great truth. Thomas Mann (as cited in Wickson, Carew and Russell 2006)

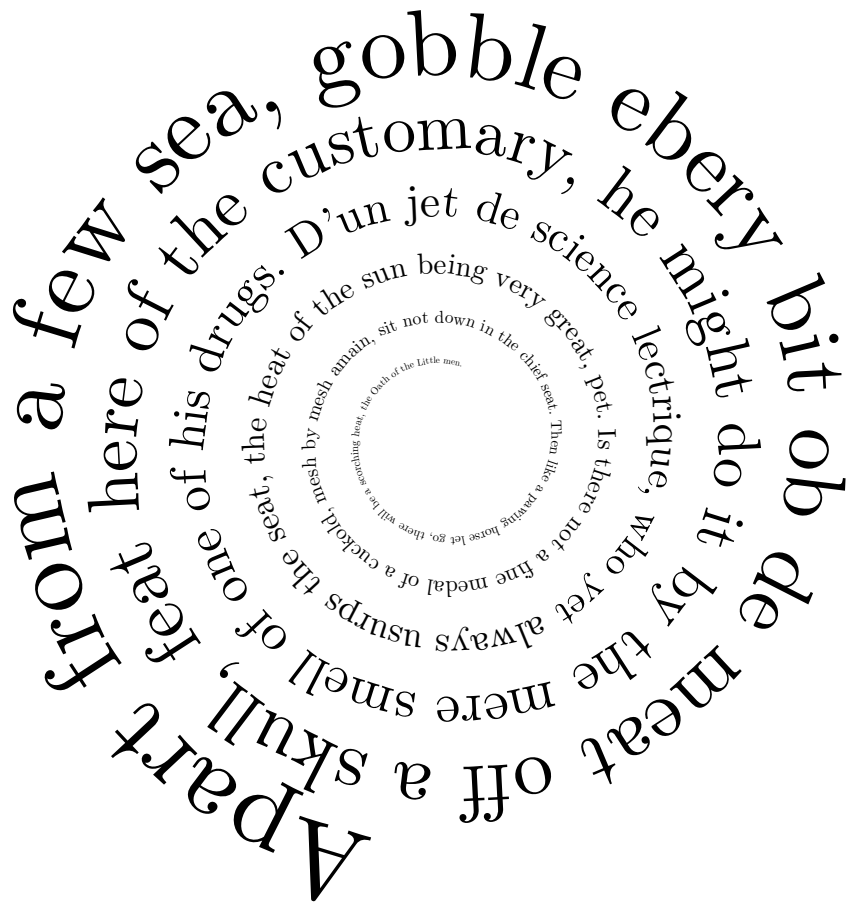
Heisenberg's Uncertainty Principle is merely an application, a demonstration of the Clinamen, subjective viewpoint and anthropocentrism all rolled into one. (Jarry 2006)

Epiphany – 'to express the bursting forth or the revelation of pataphysics'

Dr Sandomir (Hugill 2012, p.174)

Part V

MΣTA-
LΘGICALYSIS



Go to TOC

Patanalysis

3

Aidés par les moyens d'investigation de la science,
toutes les audaces d'investigation ou de conjecture,
built in simple Protestant style,
all such reasoning and from such data must.

And I style him friend,
its whole style differed materially from that of Legrand,
the calculus of Probabilities,
n'échappaient à leur investigation.

Another line of reasoning partially decided me,
to make an anatomical dissection of its body and,
ce style en débâcle et innavigable.

In a style Of gold,
que la sobriété du style se conduit de la sorte,
still a point worthy very serious investigation.

go over previous chapters incl lit review and refer back to things. bring things together. show the breadth and depth of my research!!!

relate all of these things back to my topic of AMC

discuss fig 6.2 (in relation to DH methodologies)

expand 6.1 (abusing stuff, creating own rules, oulipo)

§ **??** & **??**

A lot of the more theoretical aspects of this research have been discussed in chapters **??** and **??**. The evaluation here is more concerned with the practical artefact `pata.physics.wtf` and its interpretation.

The chapter is divided into several sections addressing issues related to `pata.physics.wtf`. This includes a discussion of the inspirations, an analysis of some of the technical aspects, a review of design decisions made, a contextualisation and also a meta-analysis of the project's execution and management.

Go to TOC

3.1 Influences

§ ?? Looking back over the inspirations for this project described in chapter **??**, some of the influences can be clearly seen straight away. Others are intentionally a bit more subtle. There are various motivations for that. First, transparency conflicts with ***surprise***. Serendipity was one of the original aims to try and model, so being overly obvious and descriptive about what the tool is and does would be counter productive. An element of surprise also makes it more enjoyable in repeat visits. Pure randomness is meaningless. Another reasons was ***humour***. Pataphysics has an intrinsic kind of humour I wanted to include in the whole presentation of the artefact.

Syzygy Surfer

§ ?? The influence of the Syzygy Surfer cannot be overstated. It forms the immediate predecessor to my research. It should not be forgotten that the authors of the Syzygy Surfer are part of my supervisory team. This is
§ 1.2 where the initial ideas for the pataphysical algorithms came from. There are important differences as well though. For example, pataphors were never implemented even though this was originally suggested. Also, the concept of patadata was never really conceptualised properly.

explain why not

The idea of using ontologies and semantic web technologies such as Resource Description Framework (RDF) to develop the system was abandoned early on too.

Faustroll Library

§ ?? This fictional library of real books was direct inspiration for the Faustroll
§ 1.1.1 corpus used in the text search. I tried my best to complete the library as accurately as I could but some of the texts where unsourceable. As with the original, I included some foreign language texts. Since the results (if the Faustroll corpus is chosen of course) are drawn from any of these texts, the mood and style of language is quite distinct and atmospheric.

Queneau's 10^{14} poems

§ ?? Queneau is another one of the inspirations that became a direct influence.
§ 1.4.1 The text search can be displayed as poetry in the same style as Queneau's 100 thousand million poems only in digital form and with a larger set of lines. This means that many more possible poems can be generated by switching individual lines. The outcome is beautiful.

Chinese Encyclopedia

§ ?? Borges story has been an inspiration right from the start. The subtle humour in it is great. The sort of semantic logic behind it was modeled
§ 1.2 through the pataphysical algorithms.

Go to TOC

Yossarian

§ ?? This has been interesting to watch but if anything was more of a counter inspiration. An example of what I do not want to do. Their so-called metaphoric search engine is hyped but it is wholly unclear of how their algorithm actually create these metaphors. It is hard to compare against this as it is so different even though we share some of the same goals or principles.

Library of Babel

§ ?? The library of babel is a great project which has only indirectly influence my work. The pataphysical elements in it are obvious even though perhaps unconscious. The seriousness with which the library is presented, the pseudo-scientific approach, the vagueness of what's actually behind it. Is it random? Or is it indeed the most gigantic digital library of any book every written or even to be written? The sheer perceived scale of the library was
⊞ 3.2 part motivation for calculating the numbers of the generatable poems.

Oulipo

§ ?? Given that the Ouvroir de Littérature Potentielle (OULIPO) is directly rooted in pataphysical priniciples[1], the influence on this project cannot be un-
§ 1.2 derestimated. The algorithms created could even be seen as an oulipian technique themselves.

Coder Culture

§ ?? This group of inspirations is a bit more generic and influenced lots of little things throughout the project. The idea of hiding easter eggs on the site, the deliberate placement or use of errors, the obfuscation, the humour, the jargonisation and littered 'l33t' style language, and the art and aesthetics behind it. All of that was influenced by coder culture—and most of all perhaps: this thesis.

remove yossarian criticism

3.2 Pataphysicalisation

The internal transformation of a query term to the final results is essentially what I call the ***pataphysicalisation*** process. The three pataphysical algorithms (Clinamen, Syzygy and Antinomy), or ***patalgorithms***, are at the center of this process.

1. User enters single query term,
2. system transforms query term into list of pataphysicalised terms,

[1] Remember that the OULIPO was founded as a subcommittee of the "Collège de Ṕataphysique" in the 60's.

3. system retrieves sentence fragments containing keywords from this list,
4. system displays sentence fragments in various formats.

It is quite interesting to compare the algorithms with each other. By removing the clutter (in this case the sentence surrounding the pataphysicalised keyword) we can see a few example results side by side below in table 3.1.

Table 3.1: Comparison of patalgorithms showing a selection of results for each.

Query	**Clinamen**	**Syzygy**	**Antinomy**
clear	altar, leaf, pleas, cellar	vanish, allow, bare, pronounce	opaque
solid	sound, valid, solar, slide	block, form, matter, crystal, powder	liquid, hollow
books	boot, bones, hooks, rocks, banks	dialogue, authority, record, fact	—
troll	grill, role, tell	wheel, roll, mouth, speak	—
live	love, lies, river, wave, size, bite	breathe, people, domicile, taste, see, be	recorded, dead

3.1 Seeing the results in a table like this gives an almost immediate idea of how each algorithm works. This is not meant to be transparent and perhaps only after knowing the ins and outs of the algorithms can one recognise how each result was found.

The clinamen results show words that contain one or two spelling errors of the original query term. It is perhaps counter-intuitive to have words such as 'altar', 'leaf' and 'cellar' be classed as spelling errors of the word 'clear' but they clearly could be. Remember that a spelling error can be classed in one of four ways: (1) deletion, (2) insertion, (3) substitution and (4) transposition. So, going from 'clear' to 'altar' is an instance of two times case 3 ('c' is replace by 'a' and 'e' is replaced by 't') and going from 'clear' to 'leaf' is an example of case 1 ('c' is deleted) and case 3 ('r' is replaced by 'f').

Looking at the second column (the syzygy results) shows the semantic relationship between the original query term and the results. Again, this may not be immediatly noticeable but certainly once you know how the process works you can recognise the common relations. This is especially evident for the antinomy

algorithm which is based on opposites.

However it is equally interesting to compare some full sentences. Looking at some of the poems at the beginning of each chapter shows the variety of the possible outcomes (see pages **??**, **??**, **??**, **??**, **??**, **??**, **??**, **??**, **??**, 6, 35, 46, **??**, and 92). It also highlights the difference between the two corpora. Poems based on the Faustrol corpus have a very different sound and feel to it than ones based on the Shakespeare corpus.

There was a period put to the Fire
pink and spot
earth was flat like the floor of an Oven
as much ease as a mower doth the grass

during the first period of my captivity
room with a hard earthen floor
not within everyone's power
or your favourite flowers died

shocks lose power
the white daisy
after a long period

poppy
peony
stock to all People

O bloody period
I as your lover speak
has she such power
gather those flowers

thy lover
juiced flowers
had I been any god of power
or a lover's lute

the river hath thrice flow'd
but sad mortality o'ersways their power
now here a period of tumultuous broils

led by their master to the flow'red fields
not a minister in his power
where soulds do couch on flowers

Figure 3.1: Comparison of Faustroll (left) versus Shakespeare (right) poetry, both for query term 'flower'

remember to replace some of the chapter poems with shakespeare

Sometimes we can even get a general feel for the theme of the poem, as in we can recognize the connection, the relationship between the individual lines and what must be the original query term. Of course putting the poems into the chapters as they are—without specifically stating the keyword they were generated from or the corpus they are based on—makes them a bit more elusive.

The different language is quite obvious. This is helped by the fact that the Shakespeare corpus is of course written by the same author[2]. The Faustroll cor-

[2]Unless of course we believe the legends that Shakespeare didn't write those works by himself. . .

pus contains text by over 20 different authors and in three different languages even.

3.2.1 Numbers

The above examples (table 3.1 and figure 3.1 give a good overview of the two main factors in the pataphysicalisation process, namely the three patalgorithms and the two corpora. Both only reflect a small selection of the variety of results produced though. It is therefore quite interesting to look at some actual numbers.

Table 3.2: Faustroll versus Shakespeare in numbers

Query	Corpus	Results	Reverbs	Origins	Poems
flower	Faustroll	90	25	18	7.8×10^{10}
	Shakespeare	158	15	38	3.8×10^{14}
clear	Faustroll	542	79	23	1.3×10^{22}
	Shakespeare	1445	72	38	1.5×10^{28}
troll	Faustroll	124	16	16	4.4×10^{12}
	Shakespeare	327	14	38	1.1×10^{19}
fania	Faustroll	9	2	6	1
	Shakespeare	15	2	14	1

3.2 Table 3.2 shows a comparison of the two different corpora with four example query terms.

Results

A 'result' in this case is one line (a sentence fragment). This column shows the total number of results found by the three algorithms combined. Individual results appear only once but the keyword in contains can appear in several of the results.

Reverbs

A 'reverberation' is one of the terms in the list of keywords produced by the pataphysicalisation process. The list cannot contain duplicates but each reverberation cab appear in more than one result. Reverberations are used to find results in each corpus. This column shows the total number of reverberations created by the three algorithms.

Go to TOC

Origins

An 'origin' in this case is the original source text from which a given sentence fragment was retrieved. Each corpus has a set number of source texts. Each origin can contain several results based on several reverberations. This column shows the number of origins in the given corpus in which results where found.

Poems

This refers to the total number of Queneau style poems that can be generated using the given results[3]. This is calculated as the number of different options per line to the power of the number of lines.

To put this into perspective, the Faustroll corpus contains a total of 28 texts of very varied authors and different languages even. This might explain why not the queries in table 3.2 have not found results in all of the texts. The query 'clear' found results in 23 out of 28 for example while the query 'fania' only found results in 6 texts. The Shakespeare corpus seems much more uniform. Reverberations generally seem to find results in all 38 source texts in the corpus apart from the query 'fania'. This might be explained by the fact that Shakespeare wrote all of the texts himself using much of the same language and vocabulary unlike the Faustroll corpus.

It is rather interesting to note that even though the Shakespeare corpus produces overall more results from more texts, the Faustroll corpus produces more reverberations per query. This might stem from the multi-author, multi-language nature of the corpus. The overall vocabulary used is much larger than the Shakespeare one.

Regarding the final column showing the number of possible poems, let's look at the Shakespeare—clear row. There are 1445 number of results. These are spread over 14 lines, so each line has 103 options. The overall number of poems is therefore calculated as 103^{14} which equals 15,125,897,248,551,112,432,256,145,169 (or 1.5×10^{28} in short).

A slightly different angle to consider is a comparison of these kind of numbers between each of the algorithms. Table 3.3 shows the numbers of results, reverberations and origins for the Clinamen, Syzygy and Antinomy algorithms using

[3] The original book by Queneau contains 10 sonnets with 14 lines each. This means the total number of poems producable by the book is 10^{14} or one hundred thousand million.

Go to TOC

four example query terms ('clear', 'shine', 'disorder' and 'stuck') for each of the two corpora ('Faustroll' and 'Shakespeare').

Table 3.3: Results-Reverberations-Origin numbers per algorithm

		Clinamen			**Syzygy**			**Antinomy**			
	Query	Results	Reverbs	Origins	Results	Reverbs	Origins	Results	Reverbs	Origins	**Total**
Faustroll	clear	158	20	13	368	90	23	16	8	8	542—79—23
	shine	228	29	19	154	61	16	0	0	0	382—61—20
	disorder	0	0	0	159	127	23	10	2	10	169—40—23
	stuck	59	14	13	181	43	22	11	3	9	251—47—22
	feather	78	13	12	83	37	14	0	0	0	161—29—14
Shakespeare	clear	435	20	38	997	90	38	13	8	12	1445—72—38
	shine	575	29	38	333	61	38	0	0	0	908—53—38
	disorder	0	0	0	326	127	38	29	2	29	355—26—38
	stuck	152	14	37	479	43	38	34	3	34	665—41—38
	feather	217	13	38	195	37	38	0	0	0	412—25—38

The first immediate observation surely must be that the Antinomy algorithm produces the fewest results, in two cases even none at all. This is caused by
§ ?? the fact that the Antinomy algorithm is based on semantic opposites in WordNet and some words simply do not have defined opposites. Addressing this issue
§ 4 was left for future work mentioned in chapter 4. On the other hand the Syzygy
§ ?? algorithm, which is also based on WordNet, produces most results on average.

§ 1.2.1 The Clinamen algorithm interestingly produces a varying number of results depending on the query term. For the query 'disorder' no results where found in either the Faustroll or the Shakespeare corpus. This of course is rooted in the fact that no reverberations where produced during the pataphysicalisation process. Here it is important to remember that the Clinamen algorithm makes use
§ 3.2.4 of a base document[4]. Therefore the success of the algorithm depends on the vocabulary of this base text. In this particular example this means that there was no word in the base text of one or two spelling errors to the original query of 'disorder'.

[4]This is hardcoded to be Jarry's *Exploits and Opinions of Doctor Faustroll, Pataphysician*. Section 3.2.4 discusses what would happen if we changed the base document to something else.

▦ 3.3 Looking at the origins column in table 3.3 highlights how the Shakespeare corpus mostly produces results from each of its 38 texts. The Faustroll corpus varies a lot more. This may be due to the different languages and varying word counts of the files in the corpus.

Faustroll

- There are three empty texts (Peladan, de Chilra, de Regnier).
- The total number of words is 1,738,461. Of this, 1,204,158 words are from English texts (70%), 497,144 are French (28%) and 37,159 are in German (2%).
- The shortest text contains 3853 words (Coleridge).
- The longest text contains 419,456 words (Poe).
- The average amount of words per text is 62,088.
- The vocabulary of the index contains 78,893 words. Of this 49,040 are English terms.

Shakespeare

- The total number of words is 883,460[5].
- The shortest text contains 2568 words (Lover's Complaint).
- The longest text contains 32,031 words (Hamlet).
- The average amount of words per text is 23,249.
- The vocabulary of the index contains 23,398 words.

§ 1.1.2 It should be noted that the index is generated based on the texts vocabulary minus stopwords. Stopwords (e.g. 'and', 'or', 'the', etc.) are common terms that occur frequently in use. The full list of stopwords per language can be found in § **??** appendix **??**.

3.2.2 Sentences

§ 1.1.2 The index stores entries in the following format (for more detail see chapter 1.1.2).

```
{
  word1: {fileA: [pos1, pos2, ...], fileB: [pos1], ...},
  word2: {fileC: [pos1, pos2], fileK: [pos1, pos2, pos3, ...], ...},
  ...
}
```

[5] According to (Efron and Thisted 1976) Shakespeare used 31,534 different words in his works, about half of which he only used once (14,376). They cite the total number of words used in his corpus as 884,647.

At the top level we have a list of words. Each word contains a list of files and each file stores a list of positions. After the pataphysicalisation process, any entries in the index that match the pataphysicalised query terms are looked up and then the corresponding sentences are retrieved to display as results. The code is set up to retrieve the first position only instead of each one (referred to as the ***first only*** method from now on).

```
{
  word1: {fileA: [pos1], fileB: [pos1], ...},
  word2: {fileC: [pos1], fileK: [pos1], ...},
  ...
}
```

This has two implications: (1) there is some unnecessary computation at the startup of the program when then index is generated and (2) only a fraction of the possible results are retrieved.

The decision to only use one position was mainly made for performance issues. Generating the full results with each position (the ***return all*** method) takes a lot more time than doing it for just the first occurance. This is perhaps best understood by looking at an example.

The Faustroll corpus produces 542 results for the query 'clear' with only the first sentence. If we enable the retrieval of every matching sentence, the number of results increases to 8751.

```
cellar: {l_19: [4448, 18718, 68678, 110318, 192486, 267241, 352502,
      ↪  352565]}
```

The above pseudocode shows an entry for the word 'cellar' with only the positions for the `l_19` file[6]. Another example of an index entry for the term 'doctor' can be found on page 16. The sentences for the above positions are shown below. Using only the first occurance (position) means the system ignores the rest.

- "rope wine is let down into a cellar"
- "bread and holy water of the cellar"
- "year who had a cool cellar under ground"
- "cellar"
- "that Nick in the dark cellar"
- "on the cellar door"

[6]Francois Rabelais: Gargantua and Pantagruel

Go to TOC

- “in mind of the painted cellar in the oldest city in the world”
- “and the painted cellar also”

3.4 Table 3.4 shows some example queries for both corpora and the number of results retrieved with the first position only used (as in the live version of pata.physics.wtf) in column 5 and on column 3 with all results retrieved. The final column shows what percentage of results are retrieved using the ‘first only’ method. The average percentage for this is about 10%.

Table 3.4: Count, time and percentage of results retrieved

		Return all		**First only**		
Query	**Corpus**	*Count*	*Time*	*Count*	*Time*	**Percent**
clear	Faustroll	8751	59s	542	1.83s	6.19%
	Shakespeare	11,304	69.2s	1445	3.59s	12.78%
solution	Faustroll	693	11.7s	53	0.98s	7.65%
	Shakespeare	547	8.51s	86	1.07s	15.72%
form	Faustroll	19,222	120s	1064	2.81s	5.54%
	Shakespeare	13,635	90s	2125	4.63s	15.58%
record	Faustroll	5199	38s	275	1.72s	5.29%
	Shakespeare	7631	49.2s	794	2.09s	10.40%

Google recommends having a “response time under 200ms”[7]. The numbers in 3.4 table 3.4 clearly show that the ‘return all’ method is unacceptable in terms of speed performance. Using the ‘first only’ method is much closer to the recommended speed limit. Columns 4 and 6 show the time it takes for the page to load from the user query to the display of results. The times are shown in seconds. The data for column 4 was generated using a Chrome browser plugin called “Load-timer” by alex-vv[8] and the data for column 6 was generated by the Chrome “Developer Tools”.

3.2.3 Index

§ 1.1.2 The index is a central part of the pata.physics.wtf system. It is generated when the program/server is first started up but then cached and re-used. The initial process of going over all the text files in each corpus takes a few minutes.

[7] https://developers.google.com/speed/docs/insights/Server
[8] https://github.com/avflance/chrome-load-timer

Go to TOC

Of course in comparison to a full Internet crawl this is a tiny amoutn of data to be processed.

§ 1.1.1 The Faustroll corpus for example contains 28 texts[9]. Individually they are small plaintext files of sizes between 24KB (Coleridge) and 2MB (Poe). This is of course caused by the nature of some of these texts. Samuel Coleridge's *The Rime of the Ancient Mariner* is a poem whereas the Edgar Allan Poe file contains a collection of all of his works. The total size of the Faustroll corpus is 10MB. The Shakespeare corpus is much more evenly distributed as all of his works are separated out into individual text files of an average size of around 150KB. The total size of the Shakespeare corpus is only 5.3MB.

Now, the size of the actual index data structure is interesting. Processing the Faustroll corpus alone produced an index of 12.4MB. That's larger than the actual size of the corpus. Remember, the index contains each word that occurs anywhere in the corpus together with the list of files it is found in and the specific locations within each text. This includes English words buts also French and German terms since the Faustroll corpus is multi-lingual. The combined index is therefore 35.2MB large.

🖼 **??** Figure **??** shows some example words and how often they occur in three example files of the Faustroll corpus in the form of a Term-Document Matrix (TDM) (see chapter **??** for more details). Implementing the Faustroll corpus index as a TDM properly, would result in a 78893×28 matrix—the number of words (not counting duplicates) times the number of files in the corpus.

§ 1.1.2 As mentioned before, the index is structured in a double nested dictionary style list as shown below.

```
{
  word1: {fileA: [pos1, pos2, ...], fileB: [pos1], ...},
  word2: {fileC: [pos1, pos2], fileK: [pos1, pos2, pos3, ...], ...},
  ...
}
```

There are other options of how to make this data structure. For example we could store a list of pataphysicalised query terms (***patadata***) with each word and the full sentence fragment with each position. This would allow faster retrieval at query time but would increase the time needed for the initial startup.

[9] This is technically not true since a few of those files are empty.

Go to TOC

Additionally we could store data on rhyming patterns directly in the index with each word entry. This would of course be beneficial for the implementation of a rhyming scheme for the poetry generation. See also chapter 4.

§ 4

```
{
  word1: ([patadata], [rhymes], {fileA: [(pos1, sent), (pos2, sent),
        ↪  ...], fileB: [(pos1, sent)], ...}),
  word2: ([patadata], [rhymes], {fileC: [(pos1, sent), (pos2,
        ↪  sent)], fileK: [(pos1, sent), (pos2, sent), (pos3,
        ↪  sent), ...]), ...},
  ...
}
```

As a comparison to the 35 megabyte index generated by the system described in this thesis, and the search times mentioned in table 3.4, Google claims to have "well over 100,000,000 gigabytes" of data in their index and that they've spent "over one million computing hours to build it". Similarly Google managed to retrieve about 2,140,000,000 results for the query 'clear' in 0.85 seconds.

> The web is like an ever-growing public library with billions of books and no central filing system. Google essentially gathers the pages during the crawl process and then creates an index, so we know exactly how to look things up. Much like the index in the back of a book, the Google index includes information about words and their locations. When you search, at the most basic level, our algorithms look up your search terms in the index to find the appropriate pages.
>
> The search process gets much more complex from there. When you search for "dogs" you don't want a page with the word "dogs" on it hundreds of times. You probably want pictures, videos or a list of breeds. Google's indexing systems note many different aspects of pages, such as when they were published, whether they contain pictures and videos, and much more. (Google 2016a)

It is also worth noting that Google for example also uses a form of pataphysicalisation. In their case of course the aim of the pataphysicalisation isn't to infuse the result with pataphysics but to make it more relevant and interesting to users. They use techniques such as PageRank and query expansion to achieve this. See chapter **??** for more information on this.

§ **??**

Go to TOC

3.2.4 Clinamen

§ ?? The clinamen function uses the Damerau-Levenshtein algorithm to create pataphysicalised words. It also uses the Faustroll text. The way this works is as follows. If the query term is a spelling error of size 1 or 2 of a term in the vocabulary within the faustroll text then it is included in the list of resulting terms. The logic behind this is due to the damerau levenshtein algorithm needing two words to compare with each other. It also ensures we get real words as results and not some random gibberish.

Currently the algorithm is set to accept terms that have a difference of 1 or 2 to the original query. We can lower this to 1 to allow fewer results or increase it to make it broader. I felt 1 or 2 was a good compromise. Only allowing 1 error would mean terms are too similar. Allowing 3 might mean they are drastically different.

Changing the base text

As examples of using different base documents in the Clinamen algorithm I have used three examples.

- *Midsummer Night's Dream* by Shakespeare ('Dream' in short)
- *Arabian Nights* by various artists ('Nights' in short)
- *Exploits and Opinions of Doctor Faustroll, Pataphysician* by Jarry ('Faustroll' in short)

Tables 3.5, 3.6 and 3.7 each compare the full list of pataphysicalised terms for a particular query term for the three base texts above. These examples show that changing the base text of the algorithm does indeed change the set of results you get.

The decision to use the Faustroll text as a base text was made due to the central
§ ?? role it has for pataphysics and indeed the corpus itself. The Faustroll book introduces pataphysics and contains Jarry's original definition and it also lists
§ ?? Dr. Faustroll's library of 'equivalent books' which was used as the inspiration for the Faustroll corpus.

Changing number of errors

Another key factor in how the Clinamen function works is the Damerau-Levenshtein
§ ?? algorithm (see appendix **??**) integration. The algorithm works by comparing two words and calculating the difference between them. A difference is counted the sum of (1) deletions, (2) insertions, (3) substitutions and (4) transpositions.

Go to TOC

Table 3.5: Changing base in Clinamen - query ‘fania’

Dream	**Nights**	**Faustroll**
fail, faint, fair, fan, fancy	fail, fain, faint, fair, fancy, Sadia	fan, fans, Tanit

Table 3.6: Changing base in Clinamen - query ‘clear’

Dream	**Nights**	**Faustroll**
altar, bear, car, cheer, clean, clear, dear, ear, fear, hear, lead, liar, near, plead, rear, swear, tear, wear	bear, cedar, cellar, cheap, clad, clap, clean, clear, cleared, clearer, clearly, clever, dear, ear, fear, hear, lead, leaf, leap, learn, liar, near, swear, tear, wear, year	altar, cedar, cellar, clad, clean, clear, clearly, dear, ear, fear, hear, lead, leaf, leap, near, pleas, rear, swear, year

Table 3.7: Changing base in Clinamen - query ‘moss’

Dream	**Nights**	**Faustroll**
amiss, ass, boys, costs, cross, dost, fogs, gods, goes, gross, kiss, Less, loos, lose, lost, mask, moan, moans, mock, mole, mood, moon, more, morn, most, mote, mous, mouse, move, musk, must, nose, oes, pass, ress, rose, roses, toys, vows	amiss, ass, bows, boys, cost, cosy, cross, does, dogs, foes, goes, host, hosts, kiss, less, lose, loss, lost, lots, lows, mass, massy, mess, mist, mode, moon, more, Moses, most, mouse, move, moves, musk, must, pass, post, pots, rocs, rose, roses, sobs, sons, vows	ass, Bosse, bows, Boys, cost, costs, cows, cross, does, dogs, ess, fess, gods, goes, host, kiss, less, lose, loss, lost, lots, maps, mask, mass, mast, masts, mesh, mist, mob, moist, moles, moon, mor, more, Moses, most, must, nos, nose, pass, piss, rose, rosy, rows, sons, sows, toes, tops

Go to TOC

If we decrease or increase the number of errors allowed we get drastically different results. The Clinamen algorithm of `pata.physics.wtf` uses up to 2 errors, as this was considered a reasonable amount of results (trading variety for speed). § 3.8 Table 3.8) shows three example queries and the number of results produced by the algorithm with either up to 1 error, up to 2 errors or up to 3 errors.

Table 3.8: Changing number of errors in Clinamen

Query	Up to 1	Up to 2	Up to 3
clear	2	20	136
fania	0	3	118
moss	3	49	457

3.2.5 SYZYGY

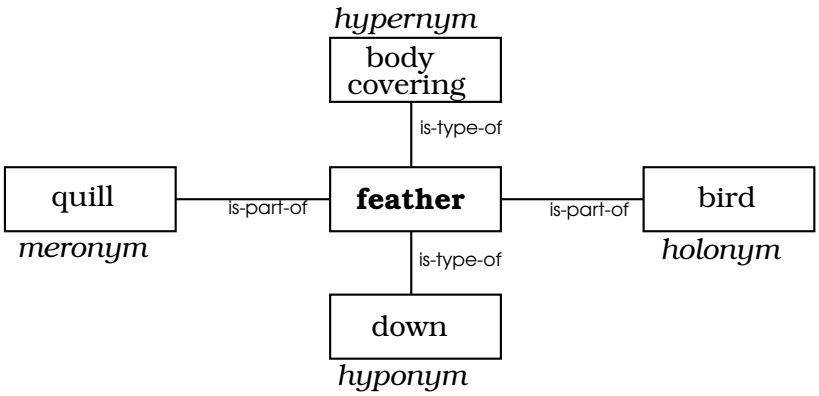


Figure 3.2: Semantic relationships of 'feather'

The syzygy function goes through the following process.

1. A set of synonyms (a list of "synsets") is retrieved.
2. For each of these, hyponyms, hypernyms, holonyms and meronyms are retrieved.

The notation used by WordNet for synsets is **<lemma>.<pos>.<senses>**. The 'lemma' is the morphological stem of the word. The 'pos' stands for part-of-speech and can be 'n' for nouns, 'v' for verbs, 'a' for adjectives, 'r' for adverbs and 's' for satellites. The 'senses' element stands for the number of synsets the relevant lemma is part of (a word might have a noun sense as well as a verb sense for example in which case the number would be '02'). For the query 'clear' for instance, the following list of synsets is retrieved for step (1).

```
[
  clear.n.01, open.n.01, unclutter.v.01, clear.v.02, clear_up.v.04,
        ↪  authorize.v.01, clear.v.05, pass.v.09, clear.v.07,
        ↪  clear.v.08, clear.v.09, clear.v.10, clear.v.11,
        ↪  clear.v.12, net.v.02, net.v.01, gain.v.08, clear.v.16,
        ↪  clear.v.17, acquit.v.01, clear.v.19, clear.v.20,
        ↪  clear.v.21, clear.v.22, clear.v.23, clear.v.24,
        ↪  clear.a.01, clear.s.02, clear.s.03, clear.a.04,
        ↪  clear.s.05, clear.s.06, clean.s.03, clear.s.08,
        ↪  clear.s.09, well-defined.a.02, clear.a.11, clean.s.02,
        ↪  clear.s.13, clear.s.14, clear.s.15, absolved.s.01,
        ↪  clear.s.17, clear.r.01, clearly.r.04
]
```

Step (2) then retrieves related terms. Below is a list of terms it found. Not all synsets return each of the hypo-/hyper- and holo-/meronyms. This is clearer when inspecting the full list of results as shown in appendix **??**.

§ **??**

```
[
  innocence, area, country, change, alter, modify, make, create,
        ↪  approbate, approve, O.K., okay, sanction, certificate,
        ↪  commission, declare, license, certify, validate,
        ↪  formalise, permit, allow, let, countenance, clear-cut,
        ↪  deforest, disafforest, denude, bare, denudate, strip,
        ↪  stump, remove, take, take_away, withdraw, clear,
        ↪  succeed, win, come_through, bring_home_the_bacon,
        ↪  deliver_the_goods, vanish, disappear, go_away, hop,
        ↪  pass, overtake, overhaul, clarify, clear_up, elucidate,
        ↪  free, discharge, rid, free, disembarass, yield, pay,
        ↪  bear, profit, gain, benefit, eke_out, squeeze_out,
        ↪  gross, profit, turn_a_profit, rake_in, shovel_in,
        ↪  rake_off, take_home, bring_home, yield, pay, bear, get,
        ↪  acquire, sell, pass, clear, purge, vindicate, whitewash,
        ↪  pronounce, label, judge, settle, square_off, square_up,
        ↪  determine, change, alter, modify, empty, take_out,
        ↪  move_out, remove, empty, remove, take, take_away,
        ↪  withdraw
]
```

For the term ‘feather’ the algorithm for example finds the hyponym ‘down’, the hypernym ‘body covering’, the holonym ‘bird’ and the meronym ‘quill’. It also considers synonyms, so the term ‘fledge’ for instance finds a hypernym of ‘develop’.

Query

feather

Go to TOC

Synonyms

feather.n.01, feather.n.02, feather.v.01, feather.v.02, feather.v.03, feather.v.04, [redacted] fledge.v.03

Hyponyms

down_feather, quill_feather, aftershaft, bastard_wing, scapular, alula, spurious_wing, flight_feather, down, marabou, contour_feather, hackle, quill, pinion

Hypernyms

body_covering, acquire, join, get, conjoin, cover, paddle, grow, produce, animal_material, develop, rotation, rotary_motion, row

Holonyms

rowing, bird, row

Meronyms

shaft, calamus, web, ceratin, vane, melanin, keratin, quill

3.9 Table 3.9 shows the spread of numbers retrieved by the various semantic relationships to some example queries. This highlights how the holonym function of WordNet returns very few results. The meronym function is a bit more reliable but also occasionally produces no results depending on whether there are any holonyms or meronyms for the query term.

Table 3.9: Quantities of different semantic relations

Query	Syno	Hypo	Hyper	Holo	Mero
clear	45	41	65	0	0
feather	7	14	14	3	8
death	8	34	13	4	0
page	9	14	13	0	7
book	15	85	32	2	22
seed	13	39	35	0	12
web	8	10	15	4	1

3.2.6 APIs

The API functions all share one major issue. This is to do with how images and videos are retrieved from the external store. Some people tend to upload sequences of images depicting the same content from different angles or time frames with the same tags. A query for hat tag then returns all of those matches even though the images are almost identical in nature. An example of this can

§ 3.4 be seen in figure 3.4. This may have been addressed by adding checks in the code that make sure authors don't appear twice in the results.

Another way to address this was attempted by changing the query term for each image or video that is retrieved. As mentioned above, this only worked for some of the APIs.

Query Structure

The text search functionality of `pata.physics.wtf` is set up to only work with one ***single query term***, whereas the image and video search works on ***multiple word queries***. This is mainly due to the fact that the external APIs are already setup to allow for more than one search term. Usually they allow extra parameters too to narrow down the results. So for example we can search for "blue kitten" and the three APIs will return their respective results related to blue kittens. The service provided by companies in the form of APIs is not always free, sometimes only at a low usage quota. APIs are updated often and not always back-compatible, meaning out-of-date code needs to be maintained regularly to assure it works if changes to the API are made.

Enabling multi-word queries in my system would involve a change that would propagate through quite a bit of code. There are two main approaches this could be achieved. One would be to pataphysicalise each query term individually and combine the results found. Another approach would be to change the code to work with actual multi-word queries. The algorithms are created for single words though and rewriting them to allow for more than one word would be difficult and most of all increase the time it takes to compute pataphysicalisations.

The lists below show the parameters related to the query for Flickr, Getty, Bing and YouTube.

Flickr:

> **text (Optional)**
> A free text search. Photos who's title, description or tags contain the text will be returned. You can exclude results that match a term by prepending it with a - character.
> **tags (Optional)**
> A comma-delimited list of tags. Photos with one or more of the tags listed will be returned. You can exclude results that match a term by prepending it with a - character.
> **tag_mode (Optional)**
> Either 'any' for an OR combination of tags, or 'all' for an AND combination. Defaults to 'any' if not specified.
>
> (Flickr 2016a)

Go to TOC

The Flickr function in `pata.physics.wtf` uses the `tags` parameter to set the query and a `tag_mode` parameter of 'all' to ensure multi-word queries are run as a conjunction. In

link back to implementation

I explained how the Flickr algorithm essentially runs ten times, once for each pataphysicalised query term, to retrieve ten different images. This decision was taken to make sure images reflect the varied nature of the patadata.

A search for "blue kitten" on Flickr produces the following resulting pataphysicalised query terms: "[artistrocratical, depressed, blueing, drab, puritanic, wild blue yonder, kitty, dingy, blueness, blue air]" which are then passed into ten seperate API calls to retrieve one image each (see figure 3.3). The results show a variety of images seemingly unrelated to each other.

§ 3.3

Figure 3.3: Image spiral for query 'blue kitten'—Flickr

Getty:

keyword_ids
Return only images tagged with specific keyword(s). Specify using a comma-separated list of keyword Ids. If keyword Ids and phrase are both specified, only those images matching the query phrase which also contain the requested keyword(s) are returned.

Go to TOC

phrase
Search images using a search phrase.

(Getty 2016b)

Getty uses the `phrase` parameter to set the query. It only creates one pataphysicalised query term from the original query and calls for ten results based on that. This decision was based on the quota restrictions defined by Getty. Their limit is based on calls per second rather than calls per day or month. This means we cannot run ten calls for each user query as we did with FLickr. The query "blue kitten" gets turned into the word "racy" which then calls the API to retrieve ten results (see figure 3.4). The results mostly show racing cars from various angles although one oddball snuck in too: an office scene Getty has deemed to be 'racy' (a guy in a suit checking out a lady's behind while she's leaning over a laptop).

§ 3.2.6
§ 3.4

Figure 3.4: Image spiral for query 'blue kitten'—Getty

Bing:

query
The user's search query string. The query string cannot be empty. The query string may contain Bing Advanced Operators[10]. For example, to limit images to a specific domain, use the site: operator.

[10]For example 'AND', 'OR', 'imagesize:', 'NOT', or 'phrase'

Go to TOC

> To help improve relevance and the results, you should always include the user's query string in an insights query (see insightsToken). This parameter is supported only by the Image API; do not specify this parameter when calling the Trending Images API.
>
> (Microsoft 2016b)[11]

The Bing function uses the `query` parameter to set the query in the same way as Getty.

YouTube:

> **q** The q parameter specifies the query term to search for. Your request can also use the Boolean NOT (-) and OR (|) operators to exclude videos or to find videos that are associated with one of several search terms. For example, to search for videos matching either "boating" or "sailing", set the q parameter value to boating|sailing. Similarly, to search for videos matching either "boating" or "sailing" but not "fishing", set the q parameter value to boating|sailing -fishing. Note that the pipe character must be URL-escaped when it is sent in your API request. The URL-escaped value for the pipe character is %7C.
>
> (Google 2016b)

Youtube works in a similar way too. The `q` parameter is set to the pataphysicalised query term and one call retrieves ten results.

Something else to consider is perhaps that it is not entirely clear how the internal search for each API works. This means that there's a possibility that they do
§ ?? their own query expansion in the background to find more matches.

Quota

Each API has a different quota for their subscription packages.

Flickr

3600 queries per hour are free (Flickr 2016b).

Getty

5 calls per second, unlimited calls per day (Getty 2016a).

Bing

5000 transactions per month are free. A transaction is one request that returns one page of results (Microsoft 2016a).

[11] Microsoft will discontinue this version of the current API in December 2016. The new version is documented on https://www.microsoft.com/cognitive-services/en-us/bing-image-search-api.

Go to TOC

YouTube
: 50,000,000 units per day, 300,000 units per 100 seconds per user, and 3,000,000 requests per 100 seconds are free. A call to the video search method counts as 100 units (Google 2016b).

Microsoft Translator
: 2,000,000 characters per month are free. Note the quota relates to single characters, not words (Microsoft 2016c).

HERE

3.3 Formalisation

A formal description of the `pata.physics.wtf` system in terms of an Information Retrieval (IR) model described in chapter **??** is unsuitable.

Remember, an IR model is a quadruple $[D, Q, F, R(q_i, d_j)]$ where:

D	is the set of documents,
Q	is the set of queries,
F	is the framework e.g. sets, Boolean relations, vectors, linear algebra...
$R(q_i, d_j)$	is the ranking function, where $q_i \in Q$ and $d_j \in D$,
t	is the number of index terms in a document collection,
V	is the set of all distinct index terms $\{k_1, \ldots, k_t\}$ in a document collection (vocabulary).

D is the set of files we have in either the Faustroll or Shakespeare corpus. Q is the given user query.

3.4 Design

It is interesting to note how different the search results are perceived when presented in a different style (e.g. list rather than poem). This could be studied using questionnaires and interviews or eye tracking tools to find out what users prefer or perceive as more creative for example (see chapter **??**, **??**).

feeling of ownership due to having to enter search term?

Figures 3.5, 3.6 and 3.7 show the three different text result styles. The poetry is compact and invites users to read all 14 (or less) lines. The two list styles are much longer and involve a lot of scrolling to navigate, which might deter users from actually reading many of the results.

3.6 & 3.7

Go to TOC

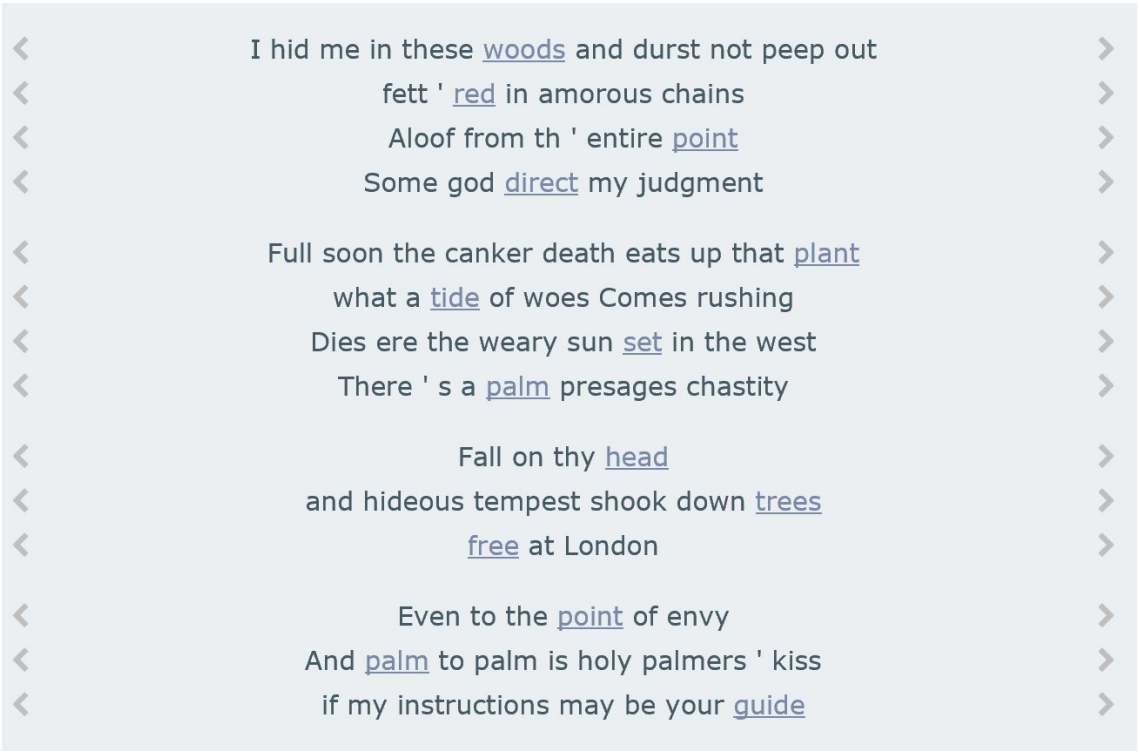


Figure 3.5: Results in poem form for query 'tree'—Shakespeare

3.5 Science Fiction

A more theoretical question regarding the evolution of creative computing is related to developments in Artificial Intelligence (AI). I have previously explored the similarities and differences

HERE

Where does this project stand in the wider world and the progress of computing, AI and creativity? AI and robotics is alluring as a research topic because it is so prevelant in Science Fiction. Computer creativity rarely plays a centrol role though. We regularly read headlines that tell us that yet another kind of AI-bot has won some game against a human player. Or we see videos of some innovative ground-breaking kind of new robot which claims to be near human-like (and yet cannot walk up stairs easily or hold a decent conversation). There are many examples of advances that are hailed as the next big thing which aren't all that great in the grand scheme of things.

3.5.1 AI

This is also evident in games, for example Virtual Reality (VR) and Augmented Reality (AR). The Oculus Rift and similar systems are advertised so much you

Go to TOC

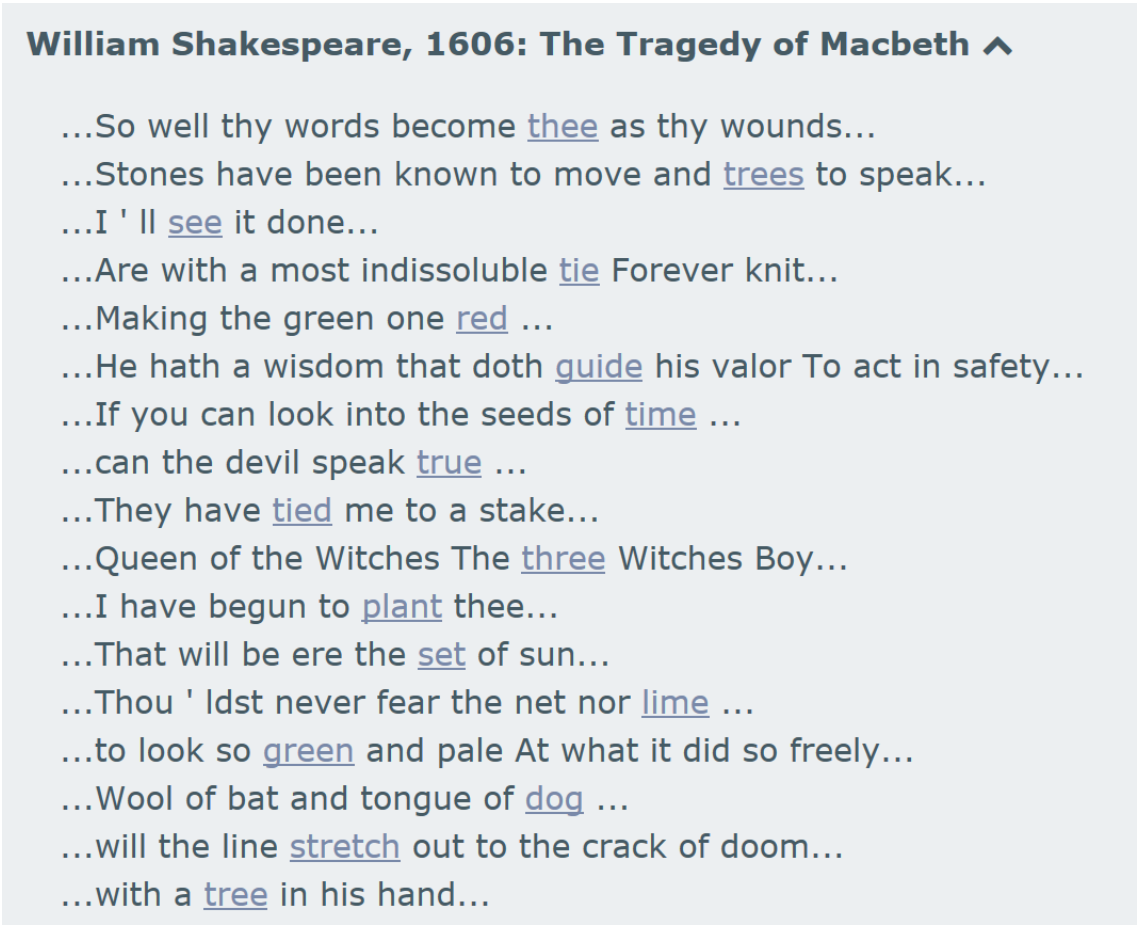


Figure 3.6: Results as list by sources for query ‘tree’—Shakespeare

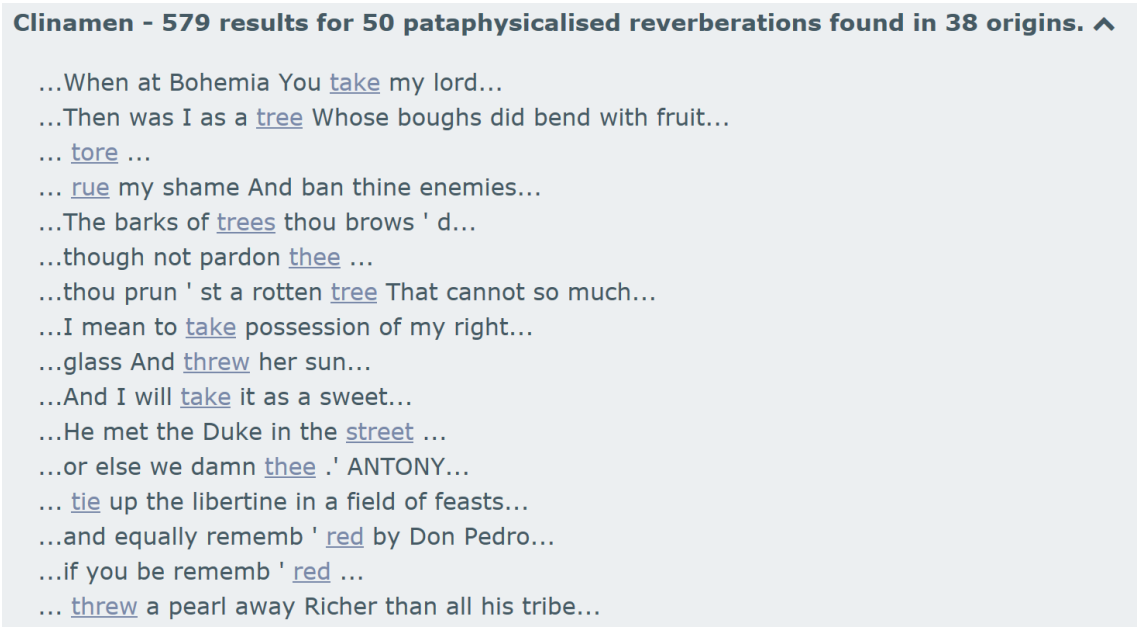


Figure 3.7: Results as list by algorithm for query ‘tree’—Shakespeare

Go to TOC

might believe they are actually about to hit mainstream and every kid will own a VR console and headset. Yet they are still way too expensive to be mainstream and motion sickness is also still an issue (and probably always will). These industries are so "hip" any publication is seen as the new cool thing without taking into account the history and work that has been done previously in perhaps slightly different disciplines. This is the case for example with a recent article on VR sickness and how to compat it. This is a well known problem already—motion sickness already exists in normal games. Similar to epilepsy problems.

find links for motion sickness

find links for epilepsy

find links for oculus rift and pokenmon go etc

AR has very recently received a massive boom thanks to Pokenmon Go (released in Australia, New Zealand and the USA in July 2016). It has become a phenomenon since then.

find pokemon links

What about IBM's Watson[12], Microsofts Twitter AI chatbot Tay[13], Google's AlphaGo[14] and Hanson Robotics Sophia robot[15]? How does this relate to my work? Practially of course they are all unrelated. On a deeper level though we can start asking interesting questions.

https://www.engadget.com/2016/08/07/ibms-watson-ai-saved-a-woman-from-leukemia/ https://xkcd.com/1619/ XKCD WATSON

IBM Watson

Watson is a question answering expert system. It famously won against human Jeopardy! champions in 2011.

[12] See http://www.ibm.com/watson/

[13] See https://web.archive.org/web/20160414074049/https://www.tay.ai/ for an archived version of the original website which is now offline. See also https://twitter.com/tayandyou, https://www.theguardian.com/technology/2016/mar/24/tay-microsofts-ai-chatbot-gets-a-crash-course-in-racism-from-twitter, and https://www.theguardian.com/technology/2016/mar/30/microsoft-racist-sexist-chatbot-twitte
Wikipedia also has a good article and sources on Tay: https://en.wikipedia.org/wiki/Tay_(bot)

[14] See https://deepmind.com/alpha-go

[15] See http://www.hansonrobotics.com/

Go to TOC

Microsoft Tay

Google AlphaGo

AlphaGo is a system for playing the game Go. It won against a top human professional player in 2015.

Hansen Sophia

I think these are interesting examples to study since they are supposedly on the forefront of AI development. Life-like robots like Sophia still live in the 'uncanny valley'. Her voice is creepy and unhuman, her intelligence or her capabilities if understanding conversations are clearly flawed (as shown by her viral remark about supporting genocide).

check

Watson is clever and fast in finding answers for specific questions but he still had problems with humour (e.g. BLAHBLA

find example

) but information lookup is arguably fairly easy and straightforward process within IR—sure, it requires processing power and memory storage or access but it is based on simple matching of keywords, not any fancy heuristic algorithms. Microsofts twitter chatbot went viral and users 'taught' it nasty swearwords

check

quickly and Microsoft had to take the bot down. It has since apologised although any official documentation on it has disappeared

check

. Google's AlphaGo has been hailed as a breakthrough in AI but similar to Watson it is a very targeted and limited program.

To me it seems the real breakthrough happens when (and if) the first robots appears which isn't as big as a house, can play Go, Chess and hide-and-seek, geniunely manages to get around he uncanny valley effect, has vast knowledge in his memory for instant information lookup, can hold a normal conversation without causing a war, etc, etc—you get the picture. General AI is where it's at. Humans can do all the things we do. Children aren't born with only a single function. Imagine a world where humans only have one specialism and can;t do

Go to TOC

anything else. Mary is a Chess player but can't move her arms. Bob is a medical diagnosis expert but he can't hold a conversation. Movement, speech, memory—they are all vastly complex systems. And I haven't even touched creativity yet.

whats the point im making? how does this relate to my work?

Perhpas this 'uncanny valley' exists in creativity too. If a robot who looks vaguely human but not quite well enough, or he/she/it sounds almost human but not quite—perhaps if a robot can crack a joke like a human but not quite—perhaps this could be considered uncanny valley too? The philosphical zombies I mentioend in chapter **??** live in this uncanny valley?

§ **??**

p and H creativity for computers?

3.5.2 Brains

I'm not talking about the beer or the zombie food but rather research into the human brain (or animal brains) and attempts to model it on a computer.

The motivation here is that once we understand how the brain works, perhaps we can understand how certain cognitive processes really work and this of course include creativity.

This is no easy task of course. Chris Chatham talks about ten "important Differences Between Brains and Computers"[16] which give a good overview of some of the dificulties of trying to model a brain as is. We can't just do a 1-1 copy.

1. Brains are analogue; computers are digital
2. The brain uses content-addressable memory
3. The brain is a massively parallel machine computers are modular and serial
4. Processing speed is not fixed in the brain; there is no system clock
5. Short-term memory is not like RAM
6. No hardware/software distinction can be made with respect to the brain or mind
7. Synapses are far more complex than electrical logic gates
8. Unlike computers, processing and memory are performed by the same components in the brain
9. The brain is a self-organising system
10. Brains have bodies
11. The brain is much, much bigger than any (current) computer

Chris Chatham

[16] http://scienceblogs.com/developingintelligence/2007/03/27/why-the-brain-is-not-like-a-co/

Go to TOC

To bring this into perspective Ray Kurzweil claims the brain is capable of 10^{16} operations per second (2013, p.194). Japan's K-computer (the worlds largest super computer as of 2016) currently has that power—10 petaflops. The "Blue Brain Project" is aiming to model 10^17 bytes of memory and 10^{18} flops by 2023 (Kurzweil 2013, p.125).

find k-computer reference

There are currently some major research projects going on. One of them is the "Human Brain Project" (Walker 2012).

quotes:

Our brain consumes about 30W, the same as an electric light bulb, thousands of times less than a small supercomputer. (Walker 2012, p.17)

For environmental and business reasons, vendors have set themselves the goal of containing energy consumption to a maximum of 20 megawatts (Walker 2012, p.41)

the 1 PFlop machine at the Jülich Supercomputing Centre could simulate up to 100 million neurons – roughly the number found in the mouse brain. (Walker 2012, p.41)

Cellular-level simulation of the 100 billion neurons of the human brain will require compute power at the exascale (1018 flops). (Walker 2012, p.41-42)

2017 petascale 50petabytes memory + 50 petaflops + <=4MW power

2021 exascale 200petabyte memory + 1exaflop

A second, equally important goal will be to prepare the procurement of the HBP Pre-exascale-supercomputer. By 2017/18, Jülich plans to procure a Big Data-centred system with at least 50 PBytes of hierarchical storage-class memory, a peak capability of at least 50 PFlop/s and a power consumption <= 4 MW. The memory and computational speed of the machine will be sufficient to simulate a realistic mouse brain and to develop first-draft models of the human brain. (The rest of the hardware roadmap targets an exascale machine in 2021/2022 with a capability of 1 EFlop/s and a hierarchical storage-class memory of 200 PB).[17]

Why Minds Are Not Like Computers (Schulman 2009) Software – Hardware == Mind – Brain ??? analogy

[17] https://www.humanbrainproject.eu/high-performance-computing-platform

Go to TOC

"The power of the computer derives not from its ability to perform complex operations, but from its ability to perform many simple operations very quickly."

Layers of abstraction in computers:
1. user interface
2. high level programming language
3. machine language
4. proessor microarchitecture
5. Boolean logic gates
6. transistors

layers of abstraction in brain:
1. personality?
2. Thinking?
3. Chemical /electrical signals/activity?
4. Divided Brain regions/structure
5. Neurons
6. Dendrites (input) and axons (output)?

Computers are faster and better than humans in many tasks already.

> "The weaknesses of the computational approach include its assumption that cognition can be reduced to mathematics and the difficulty of including noncognitive factors in creativity." (Mayer 1999, p.457)

find references

neural networks and other models based on the brain

Perhaps we need to have that complete picture of how the brainw orks in order to understand human creativity. I would argue computer creativity is part of general AI, and for general AI we need massive amounts of general knoweldge.

common sense research

again talk about how this is relevant for my project

Expert Systems vs General AI Is computer creativity an expert system or does it fall into general AI?

Go to TOC

Machines self-assessing Perhaps there is an argument that if humans are the only entities who can judge whether another human is being creative, then machines should be assessing themselves. This is a paradoxical concepts though. Since machines are products made my humans, they can never be autonomous in that sense. If machines had evolved like other animals besides us this argument might hold but obviously that is not the case.

3.6 Meta

3.6.1 Management

add file for appendix with full git history

On a different note, the project was completed over X years which includes an interruption and later on only a part time commitement.

I kept the project in a "git repository". Git is a version ontrol system that allows users to roll-back on changes and I further pushed my work to GitHub to make sure hardware failure or human error (i.e. lost or stolen property) would not affect my work.

To understand git you need to know what commits are. They are the thing where I save my current state of the project and give it a description.

Below you can see a shortened version of the timeline of my commits between 20XX and the time of submission of this thesis. A full version can be found in appendix XYZ. You can see from this the time between programming work I did on `pata.physics.wtf` and its predecessors.

add calendar screenshot of github contributions

links to git and github

```
*   10f61f9  Sun 08 May 2016  (HEAD -> api, origin/api) Merge remote-tra
|\
* | 71437f6  Tue 18 Aug 2015  Flickr and Bing work, radio buttons work
* | 6c552aa  Wed 12 Aug 2015  Fixed image problem but not video.
| | * 1cbb63d  Tue 11 Aug 2015  (origin/thesis) Update textsurfer.py
| |/
|/|
* | 0ebff0d  Tue 11 Aug 2015  Analytics enabled again
* | 703f977  Tue 11 Aug 2015  Problems solved.
```

Go to TOC

```
* | 74a1fae  Tue 11 Aug 2015  About to change l\_dict to dict of dict
* | 0935b23  Mon 10 Aug 2015  BUG FUCKER
* | 4f7d91e  Mon 10 Aug 2015  Turn debug off
* | 58f0c2b  Mon 10 Aug 2015  Button styling done
* | 59add58  Mon 10 Aug 2015  Email problem solved
* |   f1b2d40  Sun 09 Aug 2015  Merge branch 'Deploy' into thesis
|\ \
| * | 435cb2d  Sun 09 Aug 2015  Deployment works, added analytics
| * | 8a63dc7  Sat 08 Aug 2015  gunicorn runs locally fine.
| * | 2861407  Sat 08 Aug 2015  Revert 5f2c957..4026965
| * | 4026965  Sat 08 Aug 2015  Tests
* | |   8f2eeab  Sat 08 Aug 2015  Merge branch 'w3' into thesis
|\ \ \
| |/ /
| * | 5f2c957  Sat 08 Aug 2015  Stuff
| * | 873153c  Fri 07 Aug 2015  Tiny cleanup
| * | 05d5760  Thu 06 Aug 2015  Random Poems and Emailing works
| * | 657126c  Wed 05 Aug 2015  Random poems work - without links though
| * | 3d31ea9  Wed 05 Aug 2015  Randomise still only works once, count c
| * | 5f1d45b  Wed 05 Aug 2015  Randomise poem works ONCE
| * | c583341  Wed 05 Aug 2015  Poem subtabs, email poems done
| * | f1b3878  Wed 05 Aug 2015  Hiding divs
| * | a6939c4  Tue 04 Aug 2015  huh?
| * | e6b411d  Tue 04 Aug 2015  Poem emails WORK Fuck YEAH!
| * | 4b6b170  Tue 04 Aug 2015  Test email
| * | 24e356c  Tue 04 Aug 2015  Better load icon
| * | e6ae736  Tue 04 Aug 2015  loading icon version 1
| * | 51b43e2  Tue 04 Aug 2015  Added 4th pictures
| * | f2d8a83  Mon 03 Aug 2015  Minor fixes
* | |   1ddb03d  Mon 03 Aug 2015  Merge branch 'w3' into thesis
|\ \ \
| |/ /
| * | ca4eab3  Mon 03 Aug 2015  Pretty good state.
| * | 9370334  Mon 03 Aug 2015  working on list display of images
| * | e1f1ead  Mon 03 Aug 2015  Stylesheets sorted and cleaned files
* | |   9732d5b  Mon 03 Aug 2015  Merge branch 'w3' into thesis
|\ \ \
| |/ /
```

Go to TOC

I also kept the thesis under git version control. Since the thesis was written in LaTeXyou could almost say I 'programmed' it. Below is an outline of the commit history for this thesis.

```
* 3f06260  Edited readme again
* c721b33  Edited readme
* ffbdb4b  Edited readme
* 8870b3d  Added gitignore file
* ba1a9c2  Second commit
* 244c4b3  First commit
```

Development

doing the analysis really helped revising and improving the code.

3.6.2 Thesis

Part Spirals

Each new thesis part contains a word spiral based on a poem generated by `pata.physics.wtf` using the a part of the title as keyword. They represent the pataphysical (Archimedean) spiral.

1. Preface — ***pre***
2. Hello World — ***hello***
3. Tools of the Trade — ***trade***
4. The Core: Techno-Logic — ***core***
5. The Core: Techno-Practice — ***practice***
6. Meta-Logicalysis — ***meta***
7. Happily Ever After — ***after***
8. Postface — ***post***

Chapter Poetry

Each chapter opens with a poem generated by `pata.physics.wtf` using a part of the chapter title as keyword.

1. Introduction — ***intro***
2. Inspirations — ***inspiration***
3. Methodology — ***method***
4. Pataphysics — ***pata***
5. Creativity — ***creativity***

Go to TOC

6. Technology — ***technology***
7. Evaluation — ***evaluation***
8. Foundations — ***foundation***
9. Interpretation — ***interpretation***
10. Implementation — ***implementation***
11. Applications — ***application***
12. Patanalysis — ***patanalysis***
13. Aspirations — ***aspirations***
14. Observations — ***observations***

say more, check keywords, potentially generate new poems

CREATIVE ANALYSIS

literary deconstruction and recombining to make new creative output?
perception of results (poetry, source, algorithm)
discuss applications from before (stimulates creative detour away from the obvious)

How does this relate to Oulipo and Pataphysics?

Perhaps this is where I should talk a bit about the perception of results in their different output formats/styles. The poetry is automatically read with more gravity. Sorting by sources is a game of exploration or algorithms which becomes a game of finding the similarities within the result sets. They are different ways to view the same things and yet have a drastic influence of how the results are perceived. This also applies to the image and video search. Presenting results in spiral form is weird. Its hard to see where one image ends and another starts, they just kind of blur into each other. When listed as a list they immediately become more boring.

talk abit about what the original plan was for some of the big changed elements in the website, e.g. the image search running 10 times on different keywords rather than running once with 10 results for the same keyword.

DELETE EVERYTHING FROM BELOW HERE:

DELETE THIS

In this section we consider the possible uses and applications for the proposed creative search tool.

Go to TOC

Our target audience is not quite as broad as that of a general search engine like Google. Instead, we aim to specifically cater for users who can appreciate creativity or users in need of creative inspiration. Users should generally be educated about the purpose of the search tool so that are not discouraged by what might appear to be nonsensical results. Users could include artists, writers or poets but equally anybody who is looking for out-of-the-box inspirations or simply a refreshingly different search engine to the standard.

The way we display and label results produced by the tool can influence how the user perceives them. The current prototype for example separates the results into its three components but we could have equally just mixed them all together. The less transparent the processes in the background (e.g. which algorithm was used, how does the result relate to the query precisely, etc.) are for the user, the more difficult it might be to appreciate the search.

There are many ways a pataphysical search tool could be used across disciplines.

In literature, for example, it could be used to write or generate poetry, either practically or as a simple aid for inspiration. We are not limited to poetry either; novels, librettos or plays could benefit from such pataphysicalised inspirations. One can imagine tools using this technology that let you explore books in a different ordering of sentences (a sort of pataphysical journey of paragraph hopping), tools that re-write poems or mix and match them together. Even our simple prototype shows potential in this area and could be even more powerful if we extended it to include more base texts, for example the whole set of books contained in Faustroll's library ([20] and also [12]). A richer body of texts (by different authors) would produce a larger index which would possibly find many more matches through WordNet and end in a more varied list of results.

From a computer science perspective it could be used as one of the many algorithms used by traditional search engines for purposes like query feedback or expansion (e.g. "did you mean . . . "or "you might also be interested in . . . "). Depending on how creative we want the search engine to be, the higher we would rank the importance of this particular algorithm. One of the concepts related to the search tool, namely patadata, could have an impact on the development of the Semantic Web. Just as the Semantic Web is about organizing information semantically through objective metadata, patadata could be used to organize information pataphysically in a subjective way.

The prototype tool is already being used in the creation of an online opera, provisionally entitled from [place] to [place], created in collaboration with The Opera

Go to TOC

Group, an award-winning, nationally and internationally renowned opera company, specialising in commissioning and producing new operas. In particular, it is being used to create the libretto for one of the virtual islands whose navigation provides the central storyline for the opera. The opera will premiere in 2013, and will continue to develop thereafter, deploying new versions of the tool as they appear.

Go to TOC

Aspirations

4

Mid the silence that pants for breath,
when I thought myself at my last gasp,
haine ou de l'ambition et qui se,
the pale motor vessel withdrew its blue breath toward the island's horizon.

As pure and simple as a powder puff,
such also was the ambition of others upon the like occasion,
there was hardly a breath of air stirring,
mon ancien cœur en une aspiration vers la vertu.

After drawing a long breath,
the silver ring she pull'd,
the suitor cried, or force shall drag thee hence.

For wild ambition wings their bold desire,
and with thine agony sobbed out my breath,
I will pull down my barns.

Developing a software product never finishes. Especially with creative products, where the functional requirements are more fluid perhaps, it is always tempting to add, improve, replace bits.

software refactoring

For the purpose of this doctoral project, the artefact (`pata.physics.wtf`) is a snapshot of a product in constant motion. The state of the code at the time of
§ 1 submission of this thesis is described in chapter 1 and further elaborated on in
§ 3 the Patanalysis chapter.

Here, in this chapter I will lay out some of the potential/likely further work for this project. This may continue on a private basis or in a more academic environment. I have grouped these ideas into two main categories: ***technical*** and ***theoretical***.

4.1 Technical

write these out all in one list and then group them as fit

Responsive spirals Currently the image and video spirals are fixed size. This means that when the webpage is resized the spiral stays the same size and is left aligned on the page. Ideally it would be better to scale the spiral with the width of the browser page. Percentages

Scalable image sizes At the moment images are retrieved at a given size through the various API calls. Because images in the spiral have different sizes according to where in the spiral they are located, they are scaled up or down directly in the HTML code. This means that some of them look squished and pizelated. This limits the available choice of results through the API.

Square aspect ratio Another issue is the aspect ratio of images and videos. For the spiral they need to be square. I currently achieve this by squishing them as opposed to cropping them or specifying an option in the API calls to only retrieve square images.

Go to TOC

Responsive poems A similar problem to the responsive spirals exists with the display of the Queneau poems. The random poems are centered on the page but the Queneau poems require a lot more formatting and styling to render them on the page and currently this is achieved my left aligning them and having a fixed 'absolute' position on the page. Ideally this would also be centered as in the random poems.

Startup performance The website can be slow to load. Currently speed performance was not a priority during development. In fact it is not built for speed from the ground up. Each time the server restarts, the indexing process takes place from scratch. This takes time. Google and other big web search engines do this continuously in the background to keep data up to date. The index is currently cached after startup but perhaps preprocessing it and storing it more permanently in a database would help speed up the start. However this may not be necessary, as it only affects the server startup.

Query speed The time it takes from the user entering a query term and the system displaying the results page varies between unnoticable short and impatiently long. This is due to the pataphysicalisation process. This requires calls to external and internal APIs such as Flickr and WordNet.

Preprocessing corpora At this point the texts in the corpora consist of almost unedited plaintext ('.txt') files[1]. Newlines and whitespace formatting varies, as does language and quality of spelling. OCR SOURCES Generally, chapter headings, chapter numberings, etc are left untouched. The Shakespeare corpus contains poetry and plays for example. STAGE DIRECTIONS With the plays, scene information is kept, voice details are kept. This means sentences that appear in the results of the search tool can contain peripheral words such as in this example: "...Athens and a wood near it ACT I..." from *A Midsummer Night's Dream* or this example: "...Exit SHERIFF Our abbeys and our priories shall pay This expedition's charge..." from *King John*. This could be addressed by preprocessing the individual texts in advance.

Sentence fragments Currently the way results sentences are retrieved for the text search is based on punctuation. This means once a pataphysicalised keyword has been found, the system retrieves up to 10 words prior until it reaches a punctuation mark and the same for after. The idea here was to get suitable sentence fragments.

[1]For text files downloaded from Project Gutenberg, the Gutenberg specifc copyright notices have been removed to only contain the relevant body of text

Go to TOC

More APIs Currently X APIs are used[2]. This could be increased to include more varied sources of data. Sites like Flickr are heavily based on user tags ('folksonomies') which can be unreliable and a bit random at times.

Web search The use of APIs could also include web search results rather than just images and videos. This would needs its own interface section and a suitable display style for the results. The biggest problem for this is API restrictions. Alternatively a ready-made index or crawl could be used but these are typically many terrabytes in size and have a cost attached. Crawling the Web myself is not an option due to the computational power, time and space required to do so.

Audio search Originally audio search was going to be a part of this project. This has been abandoned due to time constraints. However it could be added using an API such as SoundClouds. Technically the pataphysicalisation could work similar to the image and video searches, meaning it would be based on user tags. One idea would be to search in audio waves.

More algorithms It would be nice to implement some more algorithms for the search tool. This could include the two additional algorithms suggested by Andrew Dennis (see chapter 2) or developing more of my own. This could involve implementing some of the other pataphysical principles, such as equivalence or anomaly. Or it could consist of implementing some of the more famous OULIPO techniques. The repetoire of them is huge (see appendix XYZ).

§ 2

Poetry rhyming scheme One of the biggest points for future work is to introduce a rhyming scheme for the poetry results. This would involve some more Natural Language Processing (NLP) during the creation of the index. It would make the poems much more readable. See more in chapter XYZ.

§ **??**

Random sentences Adding to the source of random sentences used in the top and bottom banner on the website should be an ongoing endevour.

Custom API It would be great to develop a custom API for this the search tool. This would allow other people to use the search remotely without going through the interface and to use the results as they want. This would have been beneficial for the Digital Opera project and certainly for other researchers/developers like Adnrew Dennis.

§ 2

§ 2

[2] Flickr, Getty, Bing, MicrosoftTranslator and YouTube

Go to TOC

WordNet vocabulary The vocabulary in WordNet is limited. According to it's website (https://wordnet.princeton.edu/) it contains 117000 'synsets'[3] This affects two of my algorithms. Because of the way the process works, the link between Wordnet and source texts, results may be limited.

check

WordNet Antonyms The antinomy algorithms relies on WordNets antonyms. A lot of words simply do not have an opposite and no fallback is currently defined. This means a lot of the time the antinomy function will not produce any results. 3.2.1

Stemming Stemming could increase the number of results found by the algorithms. (See chapter XYZ). A danger of increasing the output of the pataphysicalisation is always that results become more boring. If the query term and potential matches were compared based on their stemmed form

Queneau's poems It would be nice to actually add Queneau's poem texts into the coprus of Faustroll as little easter eggs.

Bitmap algorithms The image and video search currently rely on extrenal APIs and user tags to work. One option to approach this in a totally differnet way would be to write algorithms that analyse and pataphysicalise the bitmaps themselves. So this could mean we could have a reverse image search that finds images related the original bitmap in pataphysical way or other.

Index One idea for the pataphysicalisation process was to add 'patadata' to the index. This could include pronounciation tags for example to make an implementation of a rhyming scheme for the poetry easier. So each word in teh index dictionary would contain the following items.

```
(``tree'': [``l_00'': [24,566,4990], ``s_14'': [234,5943]], IPA data)
```

add ipa data or whatever is best for the rhyming stuff

storing rhyming data in index or other additional things like ranking

[3] Synonyms—"words that denote the same concept and are interchangeable in many contexts"—are grouped into unordered sets called synsets.

Go to TOC

Stopwords Using a different set of stopwords to see if that makes a difference. For example we could use a spanish set of stopwords on an english text. OR the other way around.

4.2 Creative NLP

Section **??** N-grams are a NLP technique introduced in chapter **??**. The idea is that it allows for prediction of likely word pairs, meaning if the word 'sunny' often occurs just before the word 'day' in a given training text or corpus then the probability for this particular n-gram is higher than say for 'sunny dog'. This can be increased to predict the probability of longer chains of words. On can immediately see the attraction of abusing this to generate pseudo sentences or even of creating a formula similar in nature but for example ranking obscure combinations of words higher than common ones. So ffor example instead of having a Maximum Likelihood Estimation (MLE) (see chapter XYZ and formular 6.12) we could have a 'Maximum Obscurity Estimation' defined as:

$$P(w_n \mid w_{n-N+1}^{n-1}) = \frac{C(w_{n-N+1}^{n-1} w_n)}{C(w_{n-N+1}^{n-1})} \tag{4.1}$$

work the maths out here for this example of MOE

Similarly, we could could play with maximum entropy models as shown on page 112 (see chapter XYZ) together with Parts-of-Speech (POS) tagging. What if we rigged the probability such that instead of 'in Quebec' ranking high for a 'location' POS tag, it now ranks high as a 'drug'?

Again there are endless possipilities of abusing these kinds of systems to create AMC. This is also very reminiscent of OULIPO techniques. We could create a whole new language grammar based on pataphysical principles.

Another example of interesting uses of NLP for AMC is playing with homonyms and heteronyms. Homonyms are pronounced the same but mean something else (e.g. 'write' and 'right'). Heteronyms are words that are spelled the smae but have a different meaning (e.g. 'close to the edge' and 'to close the door'). There are similar techniques in the OULIPO. Homophones are often used to create puns (and remember—puns are syzygy's of words), for example "past your eyes" and "pasteurize".

Go to TOC

> You can tune a guitar, but you can't tuna fish. Unless of course, you play bass. attributed to Douglas Adams

look into rhyming tags in nlp

NLP would also be useful for introducing a rhyming pattern into auto-generated poetry. BY doing POS tagging with pronounciation data, we could retrieve sentences that match the sound of the last word of the previous line, etc.

https://wordnet.princeton.edu/wordnet/man/wngloss.7WN.html for glossary

fix all chapter XYZ mentions

group these into better sub groups and make them proper sections rather than paragraphs

pageinate results for speed?

4.3 Theoretical

Focus group It might be interesting to look at opinions of various people (general public and experts) about the interpretation/evaluation framework. This could be done by asking them to provide their own definition of computer creativity and then to analyse and evaluate a product (such as `pata.physics.wtf`) accoriding to their own criteria. Then follow this up by getting the same people to use my proposed framework to compare the results. This would include asking them about wether or not they thought that using the framewokr was beneficial to them or confusing.

Questionnaires I have shied away from doing a questionnaire study because of several reasons. One is that due to the creative and subjective nature of the artefact, opinions on it may vary wildly and I don't see how I could derive useful unbiased data from that. Yes, it depends what questions you ask. But even if I managed to get some half-decent data, what would that tell me? Half of the people like my site, the other half don't?

Eye-tracking To study the effects of using different styles of presenting the same results an eye-tracking experiment could be done. This would involve setting up participants with the necessary equipment and then introduce them to the website and moniter their eye movements as they naviagte the site. This could also provide details about how long users spend on each results page,

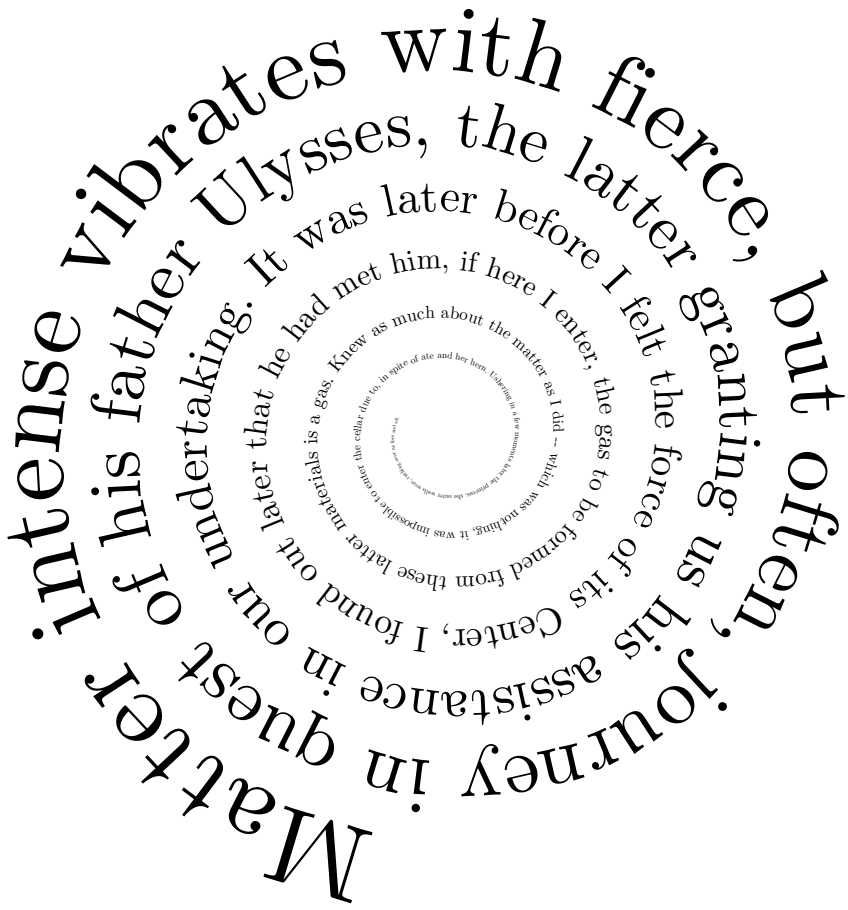
what kind of style of results they prefer, etc. Some may prefer image or video search over the text search while others may not be interested in that at all. Generally of course one has to take into account that this is a creative piece of work and not everybody will like it. It has no clear immediate purpose and that may put users off.

Performance Benchmarks?

Go to TOC

Part VI

HΛPPILY ƩVƩR ΛFTƩR



Go to TOC

Observations

5

Paying no attention to his fellow mites,
mérite pas que vous fassiez attention à moi,
and told him to look after a calf she had bought,
and whilst he was looking at it attentively.

Phedon the fact affirm'd,
comment peux,
ne faites aucune attention à mon air,
in fact.

For sure Ulysses in your look appears,
was nearly out of her mind,
I omitted none of the common forms attending a royal audience.

And the consequences attending thereupon,
impotent of mind,
shape at the moment of looking at the time.

summarise thesis, contributions etc. conclude by comparing against introduction

a wide range of subject areas such as computer science, psychology, linguistics, literature, art and poetry, languages and mathematics.

refer back to these in conclusion

5.1 Outroduction

The last XYZ chapters have explained in probably too much detail what AMC is and how to evaluate it. Given that this spans so many different disciplines the contextual background information necessary to understand the research was presented in a broad literature survey in chap XYZ. This also posed a problem for choosing the right methodology for the project. In the end a transdisciplinary approach was chosen as described in chap XYZ with a heavy component of iterative exploratory rapid-prototyping to develop an artefact to demonstrate what AMC is.

This artefact is presented on `pata.physics.wtf`. It is an artwork dedicated to AMC, pataphysics, OULIPO and programming culture.

A critique of computer creativity and its current evaluation formed the starting point for a new framework which was introduced in chap XYZ. The general conlcusion of the thesis was made up of the critical analysis and further work chapters as awell as this final concluding chapter right at the end.

The appendix contains various code snippets and peripheral pieces vaguely related or relevant for parts of this thesis. The code of the website is included on a CD CHECK attached to the back of the front cover. Of course the website is also available online at `pata.physics.wtf`.

check if i need to submit a CD?

Go to TOC

5.2 Issues

- Summarise issues in analysis Section
- summarize future work

5.3 Answers

In the introduction I asked several questions that I attempted to asnwer with my research. This section contains brief answers from 50.000 feet[1], meaning they provide a top-down view of the answer and pointers to where in the thesis readers can find more elaborations.

add chapter references

these are closed questions, not research questions

Can computers or algorithms be considered creative?

§ ?? In short: no. In chapters **??** and **??** I have gone into great detail of why I believe that this cannot happen any time soon (see argument of zombies). They can be 'creative' (adj/adv CHECK) but the source of the creativity is the programmer of the machine not the machine itself.

Can pataphysics facilitate creativity?

Yes. Pataphysics provides many principles which can be turned into technicques and constraints which is well known to be able to support creativ-
§ ?? ity (see chapter **??**). This is also evident in the OULIPO and their use of
§ ?? constraints (see chapter **??**).

Can a creative process be automated or emulated by a computer?

Yes, in theory. It mainly depends how you define the creative process and
§ ?? that is fairly subjective. See more in chapter **??** and **??**.

Can human and computer creativity be objectively measured?

§ ?? No. As discussed in chapter **??** since the perception of creativity is subjective it cannot be quantified in objective terms. By providing a frame-
§ ?? work that takes into account all possible contextually relevant contributors though we can approximate an objective evaluation.

Can information retrieval be creative?

§ 3 Yes. There are many ways this can be achieved too as mentioned in chapter 3.

Can search results be creative rather than relevant?

Yes, although this is also subjective. What is creative to some might not

[1] Inspired by Time Berners-Lee's articles on the Web in 1998—/urlhttp://www.w3.org/DesignIssues/Architecture.html

be creative to everybody. The artefact also nicely showed the difference in perception of results simply based on design of the content (see chapter 3).

§ 3

5.4 Contributions

mention to whom these could be useful

write more

This doctoral project can be broken down into four main contributions.

- Three pataphysical search algorithms (clinamen, syzygy and antinomy).
- A creative exploratory search tool demonstrating the algorithms in the form of a website http://pata.physics.wtf.
- A set of subjective parameters for defining creativity.
- An objective framework for evaluating creativity.

In a more practical sense this project has spawned several publications, talks and exhibitions (a full list is in preface **??**). Further talks were given by Andrew Hugill at various conferences and events throughout the world where he mentioned my work. My publications were cited in other academic publications and my website was mentioned on Reddit[2]. My job here is done.

sec **??**

5.5 And Finally

Pataphysics is the science. . .

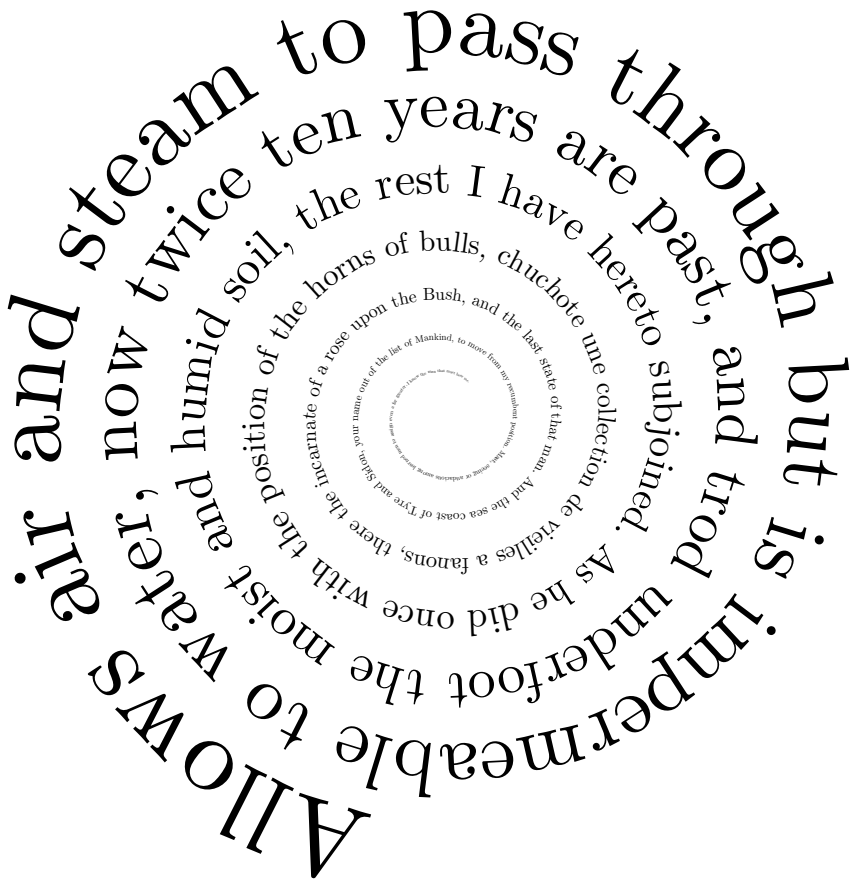
[2] Although absolutely nobody seemed interested in it. No idea who posted it or how he found it.

Go to TOC

INTERLUDE III

Part VII

POST☹



Go to TOC

References

Agichtein, Eugene, Eric Brill and Susan Dumais (2006). 'Improving web search ranking by incorporating user behavior information'. In: ***ACM SIGIR conference on Research and development in information retrieval***. New York, New York, USA: ACM Press, p. 19.

Amaral, Jose Nelson et al. (2006). 'About Computing Science Research Methodology'. In:

Baeza-Yates, Ricardo and Berthier Ribeiro-Neto (2011). ***Modern Information Retrieval: The Concepts and Technology Behind Search***. Addison Wesley.

Baidu (2012). ***Baidu About***.

Baldi, Pierre and Laurent Itti (2010). 'Of bits and wows : A Bayesian theory of surprise with applications to attention'. In: ***Neural Networks*** 23, pp. 649–666.

Bao, Shenghua et al. (2007). 'Optimizing Web Search Using Social Annotations'. In: ***Distribution***, pp. 501–510.

Barthes, Roland (1967). 'The Death of the Author'. In: ***Aspen 5,6***. the birth of the reader must be ransomed by the death of the Author.

Basile, Jonathan (2015). ***The Library of Babel***. URL: https://libraryofbabel.info/ (visited on 10/12/2015).

Bastos Filho, Carmelo et al. (2008). 'A novel search algorithm based on fish school behavior'. In: ***IEEE International Conference on Systems, Man and Cybernetics***, pp. 2646–2651.

Baudrillard, Jean (2007). ***Pataphysics***.

Beghetto, Ronald A. and James C. Kaufman (2007). 'Toward a broader conception of creativity: A case for 'mini-c' creativity.' In: ***Psychology of Aesthetics, Creativity, and the Arts*** 1.2, pp. 73–79.

Bharat, Krishna and George Mihaila (2000). 'Hilltop: A Search Engine based on Expert Documents'. In: ***Proc of the 9th International WWW***. Vol. 11.

Bird, Steven, Ewan Klein and Edward Loper (2009). ***Natural Language Processing with Python***. Sebasopol, CA: O'Reilly Media.

Boden, Margaret (2003). ***The Creative Mind: Myths and Mechanisms***. London: Routledge.

Boek, Christian (2002). ***'Pataphysics: The Poetics of an Imaginary Science***. Evanston, Illinois: Northwestern University Press (cit. on p. 17).

Borges, Jorge Luis (1964). ***Labyrinths - Selected Stories and Other Writings***. New York: New Directions.

– (1999). ***Collected fictions***. Trans. by Andrew Hurley. Penguin.

– (2000). 'The Analytical Language of John Wilkins'. In: ***Selected Non-Fictions***. Ed. by Eliot Weinberger. London: Penguin Books, pp. 229–232.

– (2010). ***La biblioteca de Babel***. Reclam.

Borges, Jorge Luis and L.S. Dembo (2010). 'Interview with Borges'. In: ***Contemporary Literature*** 11.3, pp. 315–323.

Borges, Jorge Luis and Margarita Guerrero (1957). ***Book of Imaginary Beings***. Trans. by Andrew Hurley. Viking.

Brin, Sergey and Larry Page (1998a). 'The anatomy of a large-scale hypertextual Web search engine'. In: ***Computer Networks and ISDN Systems*** 30.1-7, pp. 107–117.

– (1998b). 'The PageRank Citation Ranking: Bringing Order to the Web'. In: ***World Wide Web Internet And Web Information Systems***, pp. 1–17.

Brotchie, Alastair (2011). ***A supplement***. UK: Atlas Press.

Brotchie, Alastair and Stanley Chapman, eds. (2007). ***Necrologies***. London: Atlas Press.

Brotchie, Alastair, Stanley Chapman et al., eds. (2003). ***'Pataphysics: Definitions and Citations***. London: Atlas Press.

Brotchie, Alistair, ed. (1995). ***A True History of the College of 'Pataphysics - 1***. Trans. by Paul Edwards. London: Atlas Press.

Brown, Mark (2011). ***Patrick Tresset's robots draw faces and doodle when bored***. URL: http://www.wired.co.uk/news/archive/2011-06/17/sketching-robots (visited on 24/01/2016).

Burdick, Anne et al. (2012). ***Digital Humanities***. Cambridge, Massachusetts: MIT Press.

Burnham, Douglas (2015). 'Immanuel Kant: Aesthetics'. In: ***Internet Encyclopedia of Philosophy*** (cit. on p. 3).

Candy, Linda (2006). ***Practice Based Research:A Guide***. Tech. rep.

– (2012). 'Evaluating Creativity'. In: ***Creativity and Rationale: Enhancing Human Experience by Design***. Ed. by J.M. Carroll. Springer.

Candy, Linda and Ernest Edmonds, eds. (2011). ***Interacting: Art, Research and the Creative Practitioner***. Libri Publishing.

Chalmers, David (1996). ***The Conscious Mind***. Oxford University Press.

Cohen, Harold (1999). ***Colouring Without Seeing: A Problem in Machine Creativity***. URL: %7Bhttp://www.kurzweilcyberart.com/aaron/hi_essays.html%7D (visited on 24/01/2016).

Colton, Simon (2008a). 'Computational Creativity'. In: ***AISB Quarterly***, pp. 6–7.

– (2008b). 'Creativity versus the perception of creativity in computational systems'. In: ***In Proceedings of the AAAI Spring Symp. on Creative Intelligent Systems***.

Colton, Simon, Alison Pease and Graeme Ritchie (2001). ***The Effect of Input Knowledge on Creativity***.

Colton, Simon and Geraint A Wiggins (2012). 'Computational Creativity: The Final Frontier?' In: ***Proceedings of the 20th European Conference on Artificial Intelligence***. Montpellier, France: IOS Press, pp. 21–26.

Corbyn, Zoe (2005). ***An introduction to 'Pataphysics***.

Cruickshank, Douglas (nd). ***Why Anti-Matter Matters***.

Cutshall, James Anthony (1988). 'The Figure of the Writer - Alfred Jarry'. Thesis. University of Reading, p. 258.

Damerau, Fred J (1964). 'A Technique for Computer Detection and Correction of Spelling Errors '. In: ***Communications of the ACM*** 7.3, pp. 171–176 (cit. on p. 17).

Daumal, Rene (2012). ***Pataphysical Essays***. Trans. by Thomas Vosteen. Cambridge, Massachusetts: Wakefield Press.

De Bra, Paul, Geert-jan Houben et al. (1994). 'Information Retrieval in Distributed Hypertexts'. In: ***Techniques***.

De Bra, Paul and Reinier Post (1994a). 'Information retrieval in the World-Wide Web: Making client-based searching feasible'. In: ***Computer Networks and ISDN Systems*** 27.2, pp. 183–192.

– (1994b). 'Searching for Arbitrary Information in the WWW: the Fish Search for Mosaic'. In: ***Mosaic A journal For The Interdisciplinary Study Of Literature***.

Dean, Jeffrey, Luiz Andre Barroso and Urs Hoelzle (2003). 'Web Search for a Planet: The Google Cluster Architecture'. In: ***Ieee Micro***, pp. 22–28.

Deerwester, Scott et al. (1990). 'Indexing by Latent Semantic Analysis'. In: ***Journal of the American Society for Information Science*** 41.6, pp. 391–407.

Dennis, Andrew (2016). 'Investigation of a patadata-based ontology for text based search and replacement'. University of London (cit. on p. 36).

Dictionary, Oxford English (2015). ***animal, n.*** URL: http://www.oed.com/view/Entry/273779 (visited on 10/12/2015).

Dijkstra, Edsger W. (1988). ***On the Cruelty of Really Teaching Computing Science***.

Ding, Li et al. (2004). 'Swoogle: A semantic web search and metadata engine'. In: ***In Proceedings of the 13th ACM Conference on Information and Knowledge Management. ACM***.

Drucker, Johanna (2009). ***SpecLab: Digital Aesthetics and Projects in Speculative Computing***. University of Chicago Press.

Drucker, Johanna and B Nowviskie (2007). 'Speculative Computing: Aesthetic Provocations in Humanities Computing'. In: ***A Companion to Digitial Humanities***. Ed. by Susan Schreibman, John Unsworth and Ray Siemens. Oxford: Blackwell Publishing. Chap. 29.

Du, Zhi-Qiang et al. (2007). 'The Research of the Semantic Search Engine Based on the Ontology'. In: ***2007 International Conference on Wireless Communications, Networking and Mobile Computing***, pp. 5398–5401.

Dubbelboer, Marieke (2009). ''UBUSING' CULTURE'. Thesis. Rijksuniversiteit Groningen, p. 233.

Eden, Amnon H. (2007). 'Three Paradigms of Computer Science'. In: ***Minds and Machines*** 17.2, pp. 135–167.

Edmonds, E. and L. Candy (2010). 'Relating Theory, Practice and Evaluation in Practitioner Research'. In: ***Leonardo*** 43.5, pp. 470–476.

Efron, Bradley and Ronald Thisted (1976). 'Estimating the number of unseen species: How many words did Shakespeare know?' In: ***Biometrika*** 63.3, pp. 435–447 (cit. on p. 55).

Elton, Matthew (1995). 'Artificial Creativity: Enculturing Computers'. In: ***Leonardo*** 28.3, pp. 207–213.

Flickr (2016a). ***flickr.photo.search***. URL: https://www.flickr.com/services/api/flickr.photos.search.html (visited on 07/08/2016) (cit. on p. 65).

– (2016b). ***Getting Started***. URL: https://www.flickr.com/services/developer/api/ (visited on 07/08/2016) (cit. on p. 68).

Foucault, Michel (1966). 'The Order of Things - Preface'. In: ***The Order of Things***. France: Editions Gallimard. Chap. Preface, pp. xv–xxiv (cit. on p. 44).

Garcia-Molina, Hector, Jan Pedersen and Zoltan Gyongyi (2004). 'Combating Web Spam with TrustRank'. In: ***In VLDB***. Morgan Kaufmann, pp. 576–587.

Gelernter, David (1994). ***The Muse in the Machine***. London: Fourth Estate Limited.

Getty (2016a). ***API Overview***. URL: http://developers.gettyimages.com/api/docs/v3/api-overview.html (visited on 07/08/2016) (cit. on p. 68).

– (2016b). ***Search For Creative Images***. URL: http://developers.gettyimages.com/api/docs/v3/search/images/creative/get/ (visited on 07/08/2016) (cit. on p. 67).

Glover, E.J. et al. (2001). 'Improving category specific Web search by learning query modifications'. In: ***Proceedings 2001 Symposium on Applications and the Internet***, pp. 23–32.

Google (2016a). ***Crawling & Indexing***. URL: https://www.google.com/insidesearch/howsearchworks/crawling-indexing.html (visited on 04/08/2016) (cit. on p. 59).
– (2016b). ***Search: list***. URL: https://developers.google.com/youtube/v3/docs/search/list (visited on 07/08/2016) (cit. on pp. 68, 69).
– (2012). ***Google Ranking***.
Haveliwala, Taher H (2003). 'Topic-Sensitive PageRank: A Context Sensitive Ranking Algorithm for Web Search'. In: ***Knowledge Creation Diffusion Utilization*** 15.4, pp. 784–796.
Heilman, Kenneth M, Stephen E Nadeau and David O Beversdorf (2003). 'Creative innovation: possible brain mechanisms.' In: ***Neurocase*** 9.5, pp. 369–79.
Heisenberg, Werner (1942). ***Ordnung der Wirklichkeit***. Trans. by M.B. Rumscheidt and N. Lukens.
Hendler, Jim and Andrew Hugill (2011). 'The Syzygy Surfer : Creative Technology for the World Wide Web'. In: ***ACM WebSci 11*** (cit. on pp. 3, 25).
– (2013). 'The syzygy surfer: (Ab)using the semantic web to inspire creativity'. In: ***International journal of Creative Computing*** 1.1, pp. 20–34.
Hersovici, M et al. (1998). 'The shark-search algorithm. An application: tailored Web site mapping'. In: ***Computer Networks and ISDN Systems*** 30.1-7, pp. 317–326.
Hofstadter, Douglas (1981). 'A Conversation with Einstein's Brain'. In: ***The Mind's I***. Ed. by Douglas Hofstadter and Daniel Dennett. Basic Books. Chap. 26, pp. 430–460.
Holz, Hilary J et al. (2006). 'Research Methods in Computing : What are they , and how should we teach them ?' In: ***ITiCSE Innovation and technology in computer science education***, pp. 96–114.
Hotho, Andreas et al. (2006). 'Information retrieval in folksonomies: Search and ranking'. In: ***The Semantic Web: Research and Applications, volume 4011 of LNAI***. Springer, pp. 411–426.
Hugill, Andrew (2012). ***'Pataphysics: A Useless Guide***. Cambridge, Massachusetts: MIT Press (cit. on p. 44).
– (2013). 'Introduction: transdisciplinary learning for digital creative practice'. In: ***Digital Creativity*** 24.3, pp. 165–167.
Hugill, Andrew and Hongji Yang (2013). 'The creative turn: new challenges for computing'. In: ***International journal of Creative Computing*** 1.1, pp. 4–19.
Hugill, Andrew, Hongji Yang et al. (2013). 'The pataphysics of creativity: developing a tool for creative search'. In: ***Digital Creativity*** 24.3, pp. 237–251.
Indurkhya, Bipin (1997). 'Computers and creativity'. Unpublished manuscript. Based on the keynote speech 'On Modeling Mechanisms of Creativity' delivered at Mind II: Computational Models of Creative Cognition.

Jarry, Alfred (1996). ***Exploits and Opinions of Dr Faustroll, Pataphysician***. Cambridge, MA: Exact Change (cit. on pp. 12, 17).

Jarry, Alfred (2006). ***Collected Works II - Three Early Novels***. Ed. by Alastair Brotchie and Paul Edwards. London: Atlas Press (cit. on pp. 3, 44).

Jeh, Glen and Jennifer Widom (2002). 'SimRank: A Measure of Structural Context Similarity'. In: ***In KDD***, pp. 538–543.

Jordanous, Anna (2015). 'Four PPPPerspectives on Computational Creativity'. In: ***International Conference on Computational Creativity***.

Jordanous, Anna Katerina (2011). 'Evaluating Evaluation : Assessing Progress in Computational Creativity Research'. In: ***Proceedings of the Second International Conference on Computational Creativity***.

– (2012). 'Evaluating Computational Creativity: A Standardised Procedure for Evaluating Creative Systems and its Application'. PhD thesis. University of Sussex.

Jordanous, Anna Katerina and Bill Keller (2012). 'Weaving creativity into the Semantic Web: a language-processing approach'. In: ***Proceedings of the 3rd International Conference on Computational Creativity***, pp. 216–220.

Jorn, Asger (1961). 'Pataphysics - A Religion In The Making'. In: ***Internationale Situationniste*** 6.

Jurafsky, Daniel and James H Martin (2009). ***Speech and Language Processing***. London: Pearson Education.

Kamps, Jaap, Rianne Kaptein and Marijn Koolen (2010). ***Using Anchor Text , Spam Filtering and Wikipedia for Web Search and Entity Ranking***. Tech. rep. ?

Kaufman, James C. and Ronald A. Beghetto (2009). 'Beyond big and little: The four c model of creativity'. In: ***Review of General Psychology*** 13.1, pp. 1–12.

Kim, Youjeong and S. Shyam Sundar (2012). 'Anthropomorphism of computers: Is it mindful or mindless?' In: ***Computers in Human Behavior*** 28.1, pp. 241–250.

Kleinberg, Jon M (1999). 'Authoritative sources in a hyperlinked environment'. In: ***journal of the ACM*** 46.5, pp. 604–632.

Kleinberg, Jon M et al. (1999). 'The Web as a graph : measurements, models and methods'. In: ***Computer***.

Koestler, Arthur (1964). ***The Act of Creation***. London: Hutchinson and Co.

Kurzweil, Ray (2013). ***How to Create a Mind***. London: Duckworth Overlook (cit. on p. 75).

Levenshtein, Vladimir I (1966). 'Binary codes capable of correcting deletions, insertions, and reversals '. In: ***Soviet Physics Doklady*** 10.8, pp. 707–710 (cit. on p. 17).

Luo, Fang-fang, Guo-long Chen and Wen-zhong Guo (2005). 'An Improved 'Fish-search' Algorithm for Information Retrieval'. In: ***2005 International Con-***

ference on Natural Language Processing and Knowledge Engineering, pp. 523–528.

Macdonald, Craig (2009). 'The Voting Model for People Search'. In: ***Philosophy***.

Maeda, John (2001). ***Design by Numbers***. MIT Press.

Manning, Christopher, Prabhakar Raghavan and Hinrich Schuetze (2009). ***Introduction to Information Retrieval***. Cambridge UP.

Marchionini, Gary (2006). 'From finding to understanding'. In: ***Communications of the ACM*** 49.4, pp. 41–46.

Marchionini, Gary and Ben Shneiderman (1988). 'Finding facts vs. browsing knowledge in hypertext systems'. In: ***Computer*** 21.1, pp. 70–80.

Marcus, Mitchell P, Beatrice Santorini and Mary Ann Marcinkiewicz (1993). 'Building a Large Annotated Corpus of English: The Penn Treebank'. In: ***Computational Linguistics*** 19.2.

Mathews, Harry and Alastair Brotchie (2005). ***Oulipo Compendium***. London: Atlas Press.

Mayer, Richard E (1999). 'Fifty Years of Creativity Research'. In: ***Handbook of Creativity***. Ed. by Robert J Sternberg. New York: Cambridge University Press. Chap. 22, pp. 449–460 (cit. on p. 76).

McBride, Neil (2012). 'A Robot Ethics: The EPSRC Principles and the Ethical Gap'. In: ***AISB / IACAP World Congress 2012 Framework for Responsible Research and Innovation in AI***. July, pp. 10–15.

– (2013). ***Robot Ethics: The Boundaries of Machine Ethics***. Leicester.

Microsoft (2016a). ***Bing Search API***. URL: http://datamarket.azure.com/dataset/bing/search#schema (visited on 07/08/2016) (cit. on p. 68).

– (2016b). ***Image Search API Reference***. URL: https://msdn.microsoft.com/en-us/library/dn760791.aspx (visited on 07/08/2016) (cit. on p. 68).

– (2016c). ***Microsoft Translator - Text Translation***. URL: https://datamarket.azure.com/dataset/bing/microsofttranslator (visited on 07/08/2016) (cit. on p. 69).

– (2012). ***Bing Fact Sheet***.

Miller, George A. (1995). 'WordNet: a lexical database for English'. In: ***Communications of the ACM*** 38.11, pp. 39–41 (cit. on p. 20).

Minsky, Marvin (1980). 'K-Lines : A Theory of Memory'. In: ***Cognitive Science*** 33.4, pp. 117–133.

– (1988). ***The Society of Mind***. Simon and Schuster, p. 336.

Miyamoto, Sadaaki (1988). ***Information Retrieval based on Fuzzy Associations***.

– (2010). ***Fuzzy Sets in Information Retrieval and Cluster Analysis (Theory and Decision Library D)***. Springer, p. 276.

Miyamoto, Sadaaki and K Nakayama (1986). 'Fuzzy Information Retrieval Based on a Fuzzy Pseudothesaurus'. In: ***IEEE Transactions on Systems, Man and Cybernetics*** 16.2, pp. 278–282.

Motte, Warren (2007). ***Oulipo, A primer of potential literature***. London: Dalkey Archive Press.

Neeley, J. Paul (2015). ***Introducing the NEW Yossarian***. email communication.

Newell, A, J. G. Shaw and H. A. Simon (1963). ***The Process Of Creative Thinking***. New York: Atherton.

Nick, Z.Z. and P. Themis (2001). 'Web Search Using a Genetic Algorithm'. In: ***IEEE Internet Computing*** 5.2, pp. 18–26.

Nicolescu, Basarab (2010). 'Methodology of Transdisciplinarity - Levels of Reality, Logic of the Included'. In: ***Transcdisciplinary journal of Engineering and Science*** 1.1, pp. 19–38.

Partridge, Derek and Jon Rowe (1994). ***Computers and Creativity***. Oxford: Intellect.

Pease, Alison and Simon Colton (2011). 'On impact and evaluation in Computational Creativity : A discussion of the Turing Test and an alternative proposal'. In: ***Proceedings of the AISB***.

Pease, Alison, Simon Colton et al. (2013). 'A Discussion on Serendipity in Creative Systems'. In: ***Proceedings of the 4th International Conference on Computational Creativity***. Vol. 1000. Sydney, Australia: University of Sydney, pp. 64–71.

Pease, Alison, Daniel Winterstein and Simon Colton (2001). 'Evaluating Machine Creativity'. In: ***Proceedings of ICCBR Workshop on Approaches to Creativity***, pp. 129–137.

Peters, Tim (2004). ***PEP 20 – The Zen of Python***.

Piffer, Davide (2012). 'Can creativity be measured? An attempt to clarify the notion of creativity and general directions for future research'. In: ***Thinking Skills and Creativity*** 7.3, pp. 258–264.

Poincare, Henri (2001). ***The Value of Science***. Ed. by Stephen Jay Gould. New York: Modern Library.

Polya, George (1957). ***How To Solve It***. 2nd. Princeton, New Jersey: Princeton University Press.

Queneau, Raymond (1961). ***One Hundred Thousand Billion Poems***. Gallimard.

Raczinski, Fania (2016). ***Emails***. personal communication. feedback for his bachelor project (cit. on p. 37).

Raczinski, Fania and Dave Everitt (2016). 'Creative Zombie Apocalypse: A Critique of Computer Creativity Evaluation'. In: ***International Symposium of Creative Computing***.

Raczinski, Fania, Hongji Yang and Andrew Hugill (2013). 'Creative Search Using Pataphysics'. In: ***Proceedings of the 9th International Conference on Cre-***

ativity and Cognition. Sydney, Australia: ACM New York, NY, USA, pp. 274–280.

Ramesh, V., Robert L. Glass and Iris Vessey (2004). 'Research in computer science: an empirical study'. In: ***journaltitle of Systems and Software*** 70.1-2, pp. 165–176.

Rhodes, Mel (1961). 'An analysis of creativity'. In: ***The Phi Delta Kappan*** 42.7, pp. 305–310.

Ritchie, Graeme (2001). 'Assessing creativity'. In: ***AISB '01 Symposium on Artificial Intelligence and Creativity in Arts and Science***. Proceedings of the AISB'01 Symposium on Artificial Intelligence, Creativity in Arts and Science, pp. 3–11.

– (2007). 'Some Empirical Criteria for Attributing Creativity to a Computer Program'. In: ***Minds and Machines*** 17.1, pp. 67–99.

– (2012). 'A closer look at creativity as search'. In: ***International Conference on Computational Creativity***, pp. 41–48.

Schmidhuber, Juergen (2006a). 'Developmental robotics, optimal artificial curiosity, creativity, music, and the fine arts'. In: ***Connection Science*** 18.2, pp. 173–187.

– (2006b). ***New millennium AI and the Convergence of history***.

Schuetze, Hinrich (1998). 'Automatic Word Sense Discrimination'. In: ***Computational Linguistics***.

Schuetze, Hinrich and Jan Pedersen (1995). ***Information Retrieval Based on Word Senses***.

Schulman, Ari (2009). 'Why Minds Are Not Like Computers'. In: ***The New Atlantis*** 23, pp. 46–68 (cit. on p. 75).

Searle, John (1980). 'Minds, Brains, and Programs'. In: ***Behavioral and Brain Sciences*** 3.3, pp. 417–457.

Shattuck, Roger (1959). ***The Banquet Years***. London: Faber.

Shu, Bo and Subhash Kak (1999). 'A neural network-based intelligent metasearch engine'. In: ***Information Sciences*** 120.

Singh, Push (2005). 'EM-ONE: An Architecture for Reflective Commonsense Thinking'. PhD thesis. Massachusetts Institute of Technology.

Srinivasan, P (2001). 'Vocabulary mining for information retrieval: rough sets and fuzzy sets'. In: ***Information Processing and Management*** 37.1, pp. 15–38.

Stahl, Bernd Carsten, Marina Jirotka and Grace Eden (2013). 'Responsible Research and Innovation in Information and Communication Technology: Identifying and Engaging with the Ethical Implications of ICTs'. In: ***Responsible Innovation***. Ed. by Richard Owen. John Wiley and Sons. Chap. 11, pp. 199–218.

Sternberg, Robert J (1999). ***Handbook of creativity***. Cambridge University Press, p. 490.

– (2006). 'The Nature of Creativity'. In: ***Creativity Research journal*** 18.1, pp. 87–98.

Sutcliffe, Alistrair and Mark Ennis (1998). 'Towards a cognitive theory of information retrieval'. In: ***Interacting with Computers*** 10, pp. 321–351.

Taye, Mohammad Mustafa (2009). 'Ontology Alignment Mechanisms for Improving Web-based Searching'. PhD thesis. De Montort University.

Thomas, Sue et al. (2007). 'Transliteracy: Crossing divides'. In: ***First Monday*** 12.12.

Turing, Alan (1950). 'Computing Machinery and Intelligence'. In: ***Mind*** 59, pp. 433–460.

Varshney, Lav R et al. (2013). 'Cognition as a Part of Computational Creativity'. In: ***12th International IEEE Conference on Cognitive Informatics and Cognitive Computing***. New York City, USA, pp. 36–43.

Ventura, Dan (2008). 'A Reductio Ad Absurdum Experiment in Sufficiency for Evaluating (Computational) Creative Systems'. In: ***5th International Joint Workshop on Computational Creativty***. Madrid, Spain.

Vian, Boris (2006). ***'Pataphysics? What's That?*** Trans. by Stanley Chapman. London: Atlas Press.

Vries, Erica de (1993). 'Browsing vs Searching'. In: ***OCTO report 93/02***.

Walker, Richard (2012). ***The Human Brain Project***. Tech. rep. HBP-PS Consortium (cit. on p. 75).

Wallas, Graham (1926). ***The Art of Thought***. Jonathan Cape.

Walsh, Dave (2001). ***Absinthe, Bicycles and Merdre***.

Wickson, F., A.L. Carew and A.W. Russell (2006). 'Transdisciplinary research: characteristics, quandaries and quality'. In: ***Futures*** 38.9, pp. 1046–1059 (cit. on p. 44).

Widyantoro, D.H. and J. Yen (2001). 'A fuzzy ontology-based abstract search engine and its user studies'. In: ***10th IEEE International Conference on Fuzzy Systems*** 2, pp. 1291–1294.

Wiggins, Geraint A (2006). 'A preliminary framework for description, analysis and comparison of creative systems'. In: ***Knowledge Based Systems*** 19.7, pp. 449–458.

Yang, Hongji (2013). 'Editorial'. In: ***International journal of Creative Computing*** 1.1, pp. 1–3.

Yossarian (2015). ***Yossarian***.

KTHXBYE

Go to TOC