

修士論文 2019 年度（令和元年度）

手書きベース Wiki システムの研究

慶應義塾大学大学院 政策・メディア研究科

早川 匠

2020 年 1 月

修士論文 2019 年度（平成元年度）

手書きベース Wiki システムの研究

論文要旨

手書きメモやイラスト等のグラフィカルなデータに自在にハイパーリンクを埋め込み、それらを Wiki として利用できる「手書きベース Wiki」システムを提案する。手書きメモやイラストは広く浸透した情報の記録・表現手法であるものの、紙を前提としたフォーマットであるために参照や管理、再利用が難しいという問題が存在する。計算機上で手書きメモの作成や管理を行うツールは広く利用されているものの、これらは紙のメモやイラストの利用形態を再現したに過ぎず、この問題を本質的に解決していない。手書きベース Wiki は、ハイパーテキスト・ハイパーリンクや Wiki 等の技術の組み合わせによって、手書きでメモやイラストを描きながら、自在にハイパーリンクを埋め込んだり、ハイパーリンクによって関連する他のメモやイラストを簡単に参照できるシステムである。既存の手書きメモ・イラストの問題点を解決するだけでなく新しい活用法を提案するため、手書きベース Wiki のプロトタイプ「DrawWiki」を実装した。本論文では手書きベース Wiki としての DrawWiki の設計や評価、応用例について述べ、研究の発展性について考察する。

キーワード

手書きメモ、イラスト、Wiki、ハイパーテキスト、ユーザーインターフェース

慶應義塾大学大学院 政策・メディア研究科

早川 匠

Abstract Of Master's Thesis Academic Year 2019

A study on drawing-based wiki systems

Summary

We propose a drawing-based note-taking style where users can use handwritten objects not only for showing shapes and texts, but for linking objects just like hyperlink texts are used for linking pages. Hypertexts are widely used on wiki systems like Wikipedia, where words and phrases are used for linking pages. Although wiki systems are useful for managing a large amount of text data, it is not possible to use non-text data for linking information. It would be more useful if handwritten drawings can also be used as hyperlinks on wiki pages just like textual phrases are used for linking pages. To prove the concept of drawing-based wiki systems, we have implemented a prototype system “DrawWiki”, where arbitrary handwritten drawings can be used as links to other pages and objects.

In this paper, we describe the design, implementation, evaluations and applications of DrawWiki, and discuss the future of wiki systems where the mixture of texts and drawings are used as hyperlinks.

Keywords

Handwritten-notes, Illustration, Wiki, Hypertext, User Interface

Graduate School of Media and Governance
Keio University

Takumi Hayakawa

目 次

| | |
|----------------------------------|-----------|
| 第 1 章 序論 | 1 |
| 1.1 研究の動機 | 2 |
| 1.2 研究の目的 | 2 |
| 1.3 本論文の構成 | 3 |
| 第 2 章 研究背景 | 5 |
| 2.1 手書きメモ・イラスト | 6 |
| 2.2 手書きメモ・イラストの現状 | 6 |
| 2.2.1 iOS のメモアプリ | 6 |
| 2.2.2 Evernote | 7 |
| 2.3 手書きメモ・イラストの問題点 | 8 |
| 2.4 テキストの進化 | 8 |
| 2.4.1 Scrapbox | 8 |
| 2.5 まとめ | 9 |
| 第 3 章 設計 | 11 |
| 3.1 要件 | 12 |
| 3.2 手書きベース Wiki | 12 |
| 3.2.1 手書きベース Wiki の定義 | 12 |
| 3.2.2 DrawWiki | 12 |
| 3.2.3 DrawWiki の仕様 | 12 |
| 3.2.4 機能と使い方 | 14 |
| 第 4 章 実装 | 19 |
| 4.1 アプリケーション構成 | 20 |
| 4.2 クライアントサイド | 20 |
| 4.2.1 手書きデータの取得 | 20 |
| 4.2.2 手書きデータの描画 | 20 |
| 4.2.3 debounce された更新処理 | 21 |
| 4.2.4 リンク付要素の視覚的表現 | 21 |
| 4.3 サーバーサイド | 22 |
| 4.3.1 アプリケーションサーバー | 22 |
| 4.3.2 アセットサーバー | 23 |

| | |
|---|-----------|
| 第 5 章 利用例 | 25 |
| 5.1 グラフィカルなアイデアスケッチやメモ | 26 |
| 5.2 スケッチによるソフトウェアのモックアップの作成 | 26 |
| 5.3 インタラクティブな図解 | 26 |
| 5.4 リンクに基づいたフラットな手書きメモの管理 | 27 |
| 5.5 ナレッジの共有と共同編集 | 28 |
| 5.6 まとめ | 28 |
| 第 6 章 関連研究 | 29 |
| 6.1 手書きデータをハイパーテキストとして扱うシステム | 30 |
| 6.1.1 HyperCard | 30 |
| 6.1.2 Adobe Photoshop | 31 |
| 6.1.3 Inkscape | 31 |
| 6.2 手書きデータによって文書の参照や管理・再利用を支援するシステム | 31 |
| 6.2.1 XLibris | 32 |
| 6.2.2 InkSeine | 32 |
| 6.3 手書きスケッチをプロトタイピングに応用するシステム | 33 |
| 6.3.1 DENIM | 34 |
| 6.3.2 SILK | 34 |
| 6.4 手書きスケッチの中の要素に注釈を埋め込めるシステム | 35 |
| 6.4.1 SketchComm | 35 |
| 第 7 章 考察 | 37 |
| 7.1 評価 | 38 |
| 7.1.1 展示発表 | 38 |
| 7.1.2 意見・感想 | 38 |
| 7.1.3 筆者の運用経験 | 39 |
| 7.1.4 問題点・要望 | 39 |
| 7.2 考察 | 40 |
| 7.2.1 設計指針の妥当性 | 40 |
| 7.2.2 解決すべき課題 | 40 |
| 7.2.3 手書きメモ・イラストの問題点の検証 | 41 |
| 第 8 章 結論 | 43 |
| 8.1 研究の成果 | 44 |

| | |
|------------------|----|
| 8.2 総括 | 44 |
| 謝辞 | 45 |
| 参考文献 | 46 |

図 目 次

| | |
|--|----|
| 2.1 Apple iPad | 6 |
| 2.2 Microsoft Surface | 6 |
| 2.3 iOS のメモアプリ | 7 |
| 2.4 Evernote | 7 |
| 2.5 Scrapbox の画面 | 9 |
| 2.6 関連ページリスト | 9 |
| 3.1 DrawWiki の画面 | 13 |
| 3.2 SVG 上で表現される手書きストローク | 14 |
| 3.3 プリセットを用いて描かれた図 | 15 |
| 3.4 範囲選択ツール | 15 |
| 3.5 リンク埋め込み機能の操作画面 | 15 |
| 3.6 関連画像の表示機能 | 16 |
| 3.7 関連画像のダイアログ | 16 |
| 3.8 エキスポート機能の操作画面 | 16 |
| 3.9 Web ページに埋め込まれた手書きメモ | 16 |
| 3.10 既存の画像を表示する画面 | 17 |
| 4.1 アプリケーションの構成 | 20 |
| 4.2 PointerEvent API の概要 | 21 |
| 4.3 リンクされていない要素(上)とされている要素(下) | 22 |
| 5.1 DrawWik を用いて作成された Drawwiki のモックアップ | 26 |
| 5.2 アクションによってポップアップするダイアログの再現 | 26 |
| 5.3 既存のパーツリスト | 27 |
| 5.4 既存の部品情報リスト | 27 |
| 5.5 DrawWiki による自作整備メモ | 27 |
| 5.6 関連情報を表示した画面 | 27 |
| 6.1 エミュレータ上で再現された HyperCard | 30 |
| 6.2 操作中の XLibris | 32 |
| 6.3 InkSeine の操作画面 | 33 |
| 6.4 DENIM の画面 | 34 |
| 6.5 操作を行うタブレット機器 | 34 |

| | | |
|-----|--------------------|----|
| 6.6 | SILK の画面 | 34 |
| 6.7 | | 35 |
| 6.8 | | 35 |

表 目 次

第1章 序論

本章では本研究の動機と目的、および本論文の構成について述べる。

1.1 研究の動機

手書きによってメモやイラストを描くことは情報を記録し表現する手段として一般的である。ペンによるインターフェースを備えたタブレットデバイスが普及した現在では計算機上でも手書きメモやイラストを描くようになったが、その運用は紙の上で記録していた時代から進歩がなく、不便な点が解決されていないのが現状である。例えば描いたメモやイラストそのものを検索することができないため参照や管理が難しく、また引用する仕組みもないでの再利用が難しいという問題が存在する。

一方で手書きメモに先んじて計算機上で扱われるようになったテキストは、検索が可能なため参照や管理が行いやすく、文書作成やメール等の様々な用途で積極的に利用されるようになった。さらに Web やハイパーリンクによって目的の文書を参照したり、引用することができるようになり、またハイパーリンクを含んだ文書（ハイパーテキスト）を手軽に作成・編集できる Wiki システムによってより活発・柔軟な情報の再利用が実現された。したがって同様の技術を手書きのメモやイラストに適用することで、参照や管理、再利用が難しいという問題点を解決することができると考えられる。例えば手書きメモをハイパーテキストのように扱うシステムがあれば、メモ間のリンクに基づいた検索を行うことが可能ため参照や管理がしやすくなる。またリンクによって別のメモやイラストを引用するといった再利用も可能になる。

本研究ではハイパーテキストのような手書きのメモやイラストを自在に作成し、Wiki のように運用できるシステム「手書きベース Wiki」を開発することで手書きメモが抱える問題の解決を解決した。

1.2 研究の目的

本研究では手書きのメモを Wiki として扱えるシステム「手書きベース Wiki」を開発し、第 1.1 節で述べた手書きメモ・イラストが抱える問題を解決することを目的とする。またシステムの利用例を示し、その有用性を検証する。

1.3 本論文の構成

本論文は以下の 8 章で構成される。

第 2 章では、本研究の背景をより詳細に分析し、既存システムの問題点を整理する。

第 3 章では、本論文で提案するシステムの基本構成と使い方について述べる。

第 4 章では、本論文で提案するシステムの詳細な実装について述べる。

第 5 章では、本論文で提案するシステムの利用例を紹介する。

第 6 章では、関連する研究を紹介し、それらの特徴や本研究との関連を述べる。

第 7 章では、筆者による運用経験やユーザーからのフィードバックをまとめ、本論文で提案するシステムの有効性と問題点について述べる。

最後に、第 8 章で本論文のまとめと結論を述べる。

第2章 研究背景

本章では手書きメモ・イラストを扱う既存のシステムの現状と、その問題点を整理する。

2.1 手書きメモ・イラスト

手書きによるメモやイラストは情報を記録・表現する手法として広く普及している。例えば物体のスケッチ等の視覚的な情報やアイデアのような文章のみでは表現しづらい構造を持つ概念等を表現する場合は、文字と図を自在に混合させることができる手書きメモの方が、テキストのみを使用したメモよりも柔軟に記録できる。さらに手書きメモの作成にあたってはペンさえ握れれば良く、タッチタイピング等の特別なスキルも必要としないためテキスト入力と比較してより多くの人々にとって利用しやすい表現手段であると言える。

また iPad(図 2.1) や Surface(図 2.2) 等のペンによるインターフェースを備えたタブレットデバイスが普及した現在では、テキストと同じく計算機上でも手書きのメモやイラストが描かれるように変化した。これにより手書きのメモやイラストを描く際に Undo/Redo やコピー&ペースト等の編集支援機能を利用できるようになり、利便性が向上した。一方でテキストと異なり手書きのデータそのものを検索することができないため、作成した手書きメモの運用については不便な点が存在する。

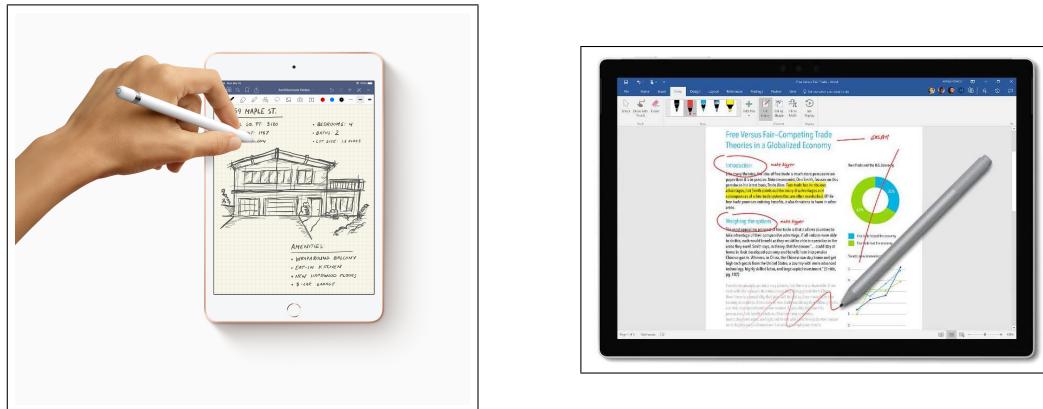


図 2.2: Microsoft Surface

図 2.1: Apple iPad

2.2 手書きメモ・イラストの現状

計算機上で手書きメモ・イラストを扱う主要なシステムを紹介し、計算機における手書きメモの現状を解説する。

2.2.1 iOS のメモアプリ

Apple¹の iOS デバイスにはメモアプリケーションが標準でインストールされており、指や Apple Pencil を用いて素早く手書きメモを取ることができる。一方で描いた手書きメモは単なる画像として扱われ、それを後から検索する機能は存在しないため 作成した手書き

¹<https://www.apple.com/>

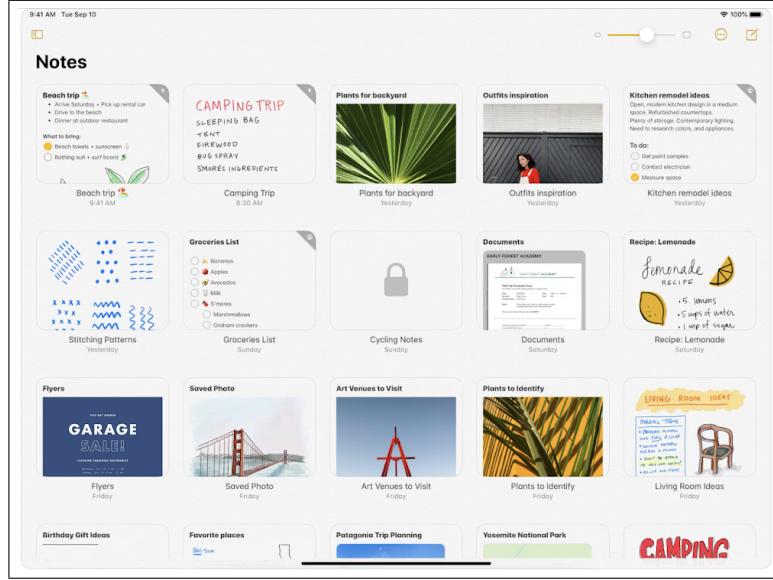


図 2.3: iOS のメモアプリ

メモを参照可能な状態にするにはルールに基づいてファイル名を決めたり、図 2.3 のようにフォルダに分類して保存する等の工夫が求められる。

2.2.2 Evernote

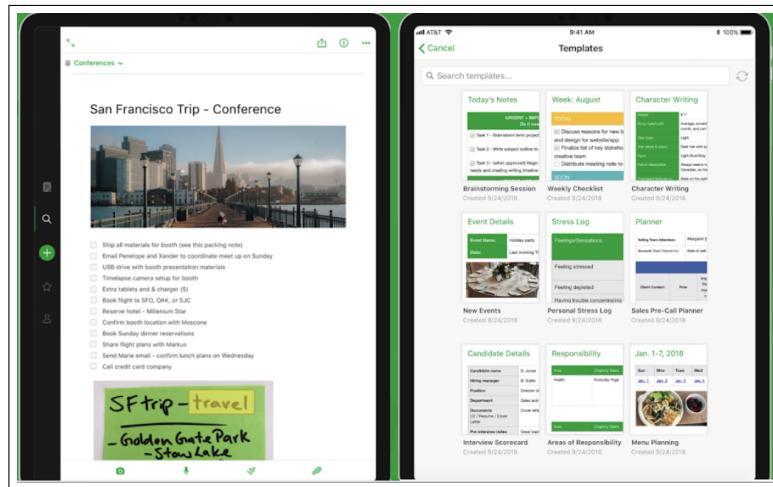


図 2.4: Evernote

Evernote Corporation²が開発する Evernote(図 2.4) も指やスタイルスペンによって手書きのメモやスケッチを作成できる。また iOS 標準のメモアプリと同様にノートブックと呼ばれる分類用のフォルダを備えるほか、ラベルによってメモを分類することもできる。さら

²<https://evernote.com/intl/jp/about>

に手書きメモ内の文字を認識し、全文検索によって手書きメモを検索する機能を備えているが、検索対象は認識できた手書き文字のみであり、図形や描いたものの形状等のグラフィカルなデータから手書きメモを検索することはできない。

またいずれのメモアプリケーションも、作成した手書きのメモをそのアプリケーション以外のシステムで再利用することが難しく、相互運用性に問題がある。例えば作成した iOS のメモアプリや Evernote で作成した手書きメモを Web サイトに埋めこむことはできない。いずれのアプリも手書きの部分を独立した画像ファイルとして出力できるが、その場合はメモに関するフォルダやラベル等の分類情報も欠落してしまう。

2.3 手書きメモ・イラストの問題点

現状の手書きメモ・イラストが抱える問題点を整理する。

1. 検索ができないため参照や管理が面倒

テキストと異なり手書きメモ・イラストはそのものを検索することができないため参照しづらい。参照可能な状態にしておくためにはフォルダやラベルによる分類等の管理が必要とされる。

2. 再利用が難しい

Web サイトへの埋め込み等、作成したメモアプリケーション以外で手書きメモを再利用することが難しい。

2.4 テキストの進化

手書きに先んじて計算機上で利用されているテキストは、文字から内容を検索することができるため文書の参照や管理が格段に行いやすく、あらゆる文書の作成やメール等の連絡手段が電子化されたテキストに置き換えられるようになった。また Web やハイパーリンク等の技術によってより参照しやすくなったほか、別の文書を引用する等の再利用が可能になった。さらにハイパーリンクを含んだ文書(ハイパーテキスト)を手軽に作成・編集できる Wiki[8] が登場したことにより柔軟かつ活発なテキストによる知見の共有や情報の再利用が実現された。

2.4.1 Scrapbox

Wiki システムの例として Gyazz[16] をベースに開発され、Nota³社によって運営されている Scrapbox⁴を挙げる(図 2.5)。Scrapbox は [] を基本としたシンプルな記法と WYSIWYG エ

³<https://www.notainc.com/ja>

⁴<http://scrapbox.io/>

ディタによってページ間のリンクや外部リンクを含んだ画像や動画を簡単に記述することができる。また図 2.6 のようにページの下部に

- 別ページへのリンク
- 別ページからのリンク
- リンク先ページがリンクしているページ

等のリンク情報に基づいた関連ページを推薦し表示する機能を備えている。Scrapbox には分類用のフォルダ等の機能は存在せず全てのページがフラットに管理されるが、リンクによって関連するページが自動的に分類されるため管理のための特別な工夫を必要とせずに多数のページを運用することができる。

このように Wiki システムによって文書の参照や管理が容易になり、さらなる情報の再利用が可能になった。

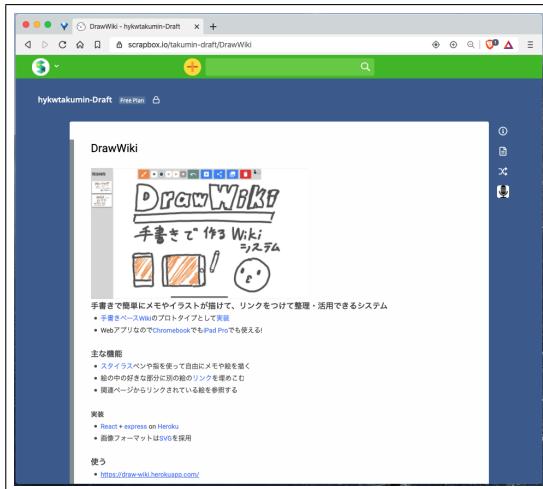


図 2.5: Scrapbox の画面

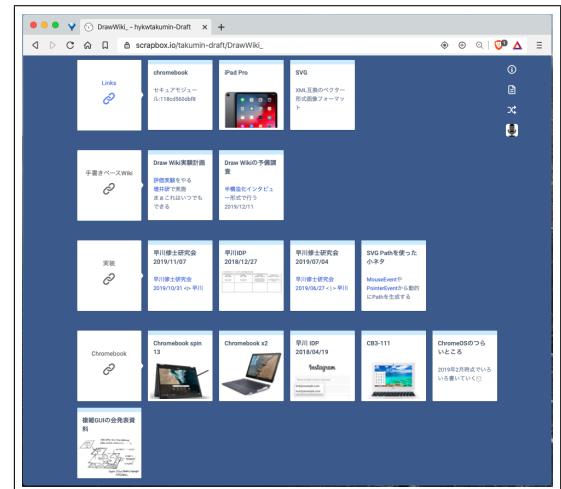


図 2.6: 関連ページリスト

2.5 まとめ

手書きメモ・イラストは広く浸透した情報の記録・表現手法であるものの、参照や管理には問題があり、再利用も制限されている。一方でテキストは計算機上でも積極的に応用されており、Wiki 等の技術によって文書の参照や管理が容易になったことからさらなる情報の再利用が実現された。したがって同様の技術を手書きメモ・イラストに適用することができれば、問題を解決できると考えられる。次章では手書きメモをハイパーテキストとして扱い、Wiki のように運用できるシステム「手書きベース Wiki」を提案し、その設計について述べる。

第3章 設計

本章では手書きベース Wiki の要件と設計について述べる。

3.1 要件

前章で示した手書きメモ・イラストを扱う既存のシステムの問題点を踏まえて、本システムの要件を整理する。

1. 簡単に手書きメモのメモやイラストが作成・編集できる
気軽に手書きメモ・イラストを取ることができること。
2. 作成した手書きのメモやイラストを簡単に参照したり、再利用したりできる
画像の内部に対してハイパーリンクを設定でき、関連する画像を参照することができる。

3.2 手書きベース Wiki

本研究で提案する手書きベース Wiki の基本構成及び使い方を解説する。

3.2.1 手書きベース Wiki の定義

- 手書きメモ・イラストを作成・編集できる
タッチやスタイルスペンに対応したエディタを備え、手書きメモ・イラストを手軽に作成・編集することができる。
- 手書きメモ・イラストにハイパーリンクを追加できる
作成した手書きメモ・イラスト同士の相互リンクを簡単な操作で定義できる。
- 関連する手書きメモ・イラストを参照できる
メモに追加されたリンク情報を元に、関連するメモ・異羅臼とを推薦し、一覧できるよう表示する。

この要件を満たす手書きベース Wiki のプロトタイプとして DrawWiki を開発した。

3.2.2 DrawWiki

DrawWiki は本研究において開発した、手書きベース Wiki のコンセプトを元にしたプロトタイプとなるアプリケーションである。(図 3.1) 本章ではその主要な機能を使い方とともに解説する。

3.2.3 DrawWiki の仕様

DrawWiki における手書きメモ・イラストは SVG[2] 形式で記録される。SVG は XML¹をベースとする画像ファイルフォーマットであり、手書きメモ・イラストをハイパーテキスト

¹<https://www.w3.org/XML/>



図 3.1: DrawWiki の画面

として扱う上で有利な以下のような特徴を持つ。

- グラフィカルな表現を前提に設計されている
SVG には曲線等を表現する Path 要素や閉じた図形を表現する Polygon 要素等の仕様が標準で備わっている。スタイラスから得られるデータを Path 要素の属性として定義することで手書きのストロークを表現することができる。(例えばソースコード 3.1 で図 3.2 のように表現できる)
- 構造を保持できる
ラスターイメージと異なり SVG の実体は構造化されたテキストファイルであり、線分や点はピクセルではなく独立した要素として記述される。これにより書き順や編集履歴等の構造も保持され、再編集性が高い。
- ハイパーリンクを埋め込める
SVG では XLink²形式のハイパーリンクを任意の要素に埋め込むことができるため画像の中の個別の要素に対して複数のリンクを定義することができる。
- Web 標準の技術である
SVG は特定の企業の製品ではなく、その仕様は全て公開されている。また作成や表示に特別なソフトウェアを必要とせず、ブラウザがあれば閲覧することができる。

ソースコード 3.1: 図 3.2 の実体

```

1 <path stroke-linejoin="round" stroke-linecap="round" stroke="#585858"
2   stroke-width="10" class="" pointer-events="auto" fill="rgba(0,0,0,0)"
3   id="1072-528" d="M_919_564_,L_919_563_,L_919_562_,L_919_560_,L_919_559_
...L_1064_520_"></path>

```

²<https://www.w3.org/TR/xlink/>

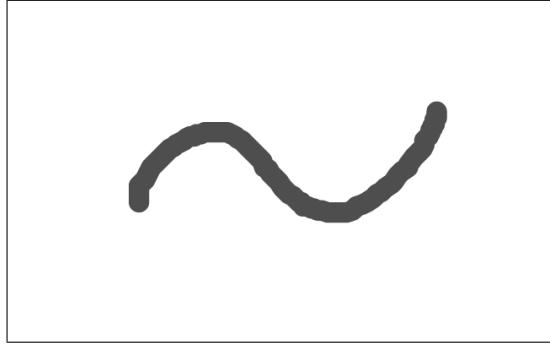


図 3.2: SVG 上で表現される手書きストローク

3.2.4 機能と使い方

(1) 手書きメモ・イラストの作成・編集

画面中央部分に自由に手書きができるキャンバスが配置されており、ここに描いたものが自動的に保存・アップロードされる。一度アップロードしたメモ・イラストには一意な URL が割り振られるため Web を通じて他のユーザーが閲覧し、編集することもできる。

既存の手書きメモアプリケーションと同様に、Redo 等の編集支援機能を利用可能であるほか、DrawWiki では金箱らによる Interactive Sketch[15] に基づいたコンパクトなブラシプリセット構成を採用している。このスケッチ技法は以下の 4 種類のブラシを用いる

1. 通常のペンで全体を描く
2. 影や質感をグレーで表現する
3. 輪郭を太い線で縁取る
4. 特徴的な部分、機能を有する箇所をキーカラーでハイライトする

これにより Wiki として活用する上で有利な以下の特徴を持つ(図 3.3)。

- 太線で図のアウトラインが強調されるためサムネイル状態での視認性が確保される
- キーカラーによる強調表示でアイデアが伝わりやすくなる
- ブラシによって書き方がある程度統一され、スケッチ技量の巧拙を吸収できる



図 3.3: プリセットを用いて描かれた図

(2) ハイパーリンク埋め込み機能



図 3.4: 範囲選択ツール

図 3.5: リンク埋め込み機能の操作画面

範囲選択ツールを用いてハイパーリンクを埋め込みたい要素を選択することができる。要素を選択して”他の図とリンクボタン”を押すと今まで作成したメモ・イラストの一覧がダイアログで表示され、その中からリンクさせたい図を選ぶとその図へのハイパーリンクが要素に埋め込まれる。

(3) 関連画像の表示機能

エディタの横に位置する”関連画像ビュー”には、

- 現在表示中メモがのリンクしている画像



図 3.6: 関連画像の表示機能

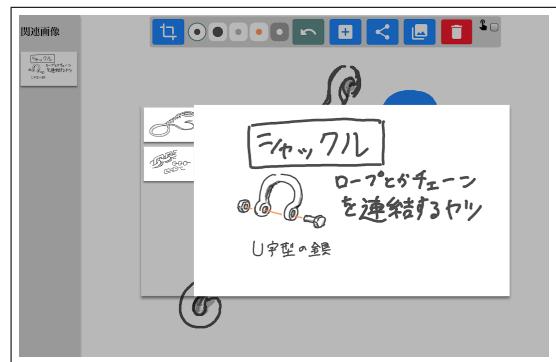


図 3.7: 関連画像のダイアログ

- 現在表示中メモをリンクしている画像

等のリンクに基づいた関連画像のサムネイルが表示される(図3.6)。また、サムネイルを選択するとリンクされている要素が強調表示され、対応関係が可視化される。さらにリンク付要素をクリックすると、リンク先の画像がダイアログで表示される(図3.7)。このダイアログの内部にも関連画像ビューが内蔵されており、

当該ハイパーイラストがさらに別の画像をイラストをリンクしている場合はその画像もダイアログに表示される。このようにリンクに基づいて関連するイラストを表示し、参照可能にする仕組みが備わっている。

(4) エキスポート・共有機能

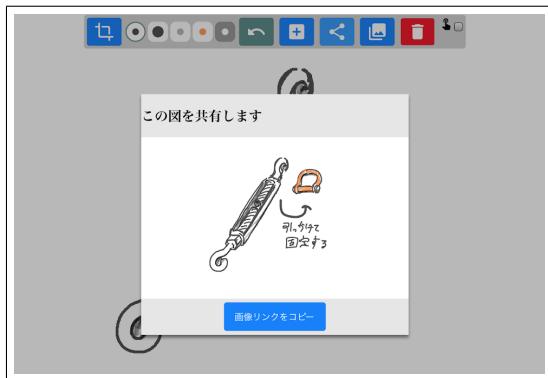


図 3.8: エキスポート機能の操作画面



図 3.9: Web ページに埋め込まれた手書きメモ

エキスポート機能を利用すると、手書きメモ・イラストのファイルそのもの(SVG)のURLを取得することができる。このURLをimg要素³やiframe要素⁴のsrc属性やObject要素⁵のdata属性に指定する事が可能で、図3.9のように他のWebサイトにも埋め込んで利用することができる。画像としてエキスポートした状態でも埋め込んだハイパーリンクは有効であり、埋め込まれた要素をクリックするとリンクされた別のメモにアクセスすることができる。

(5) 画像一覧機能

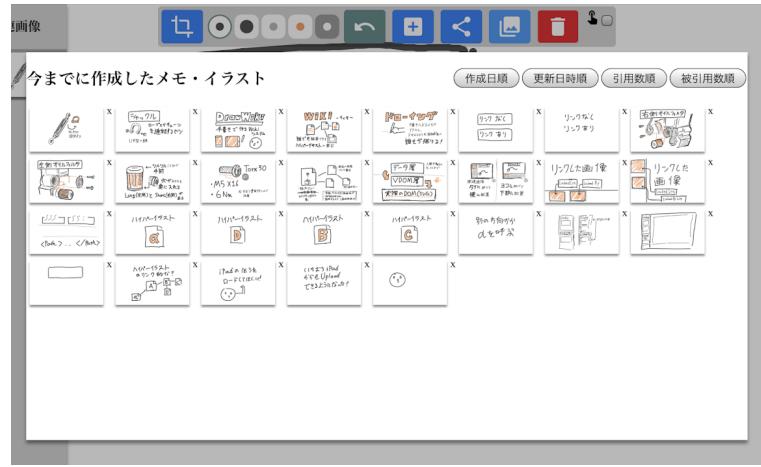


図 3.10: 既存の画像を表示する画面

DrawWikiで作成した画像は一覧画面から表示することができる。デフォルトでは更新された日時が新しい画像から順に表示しているが、作成日順、引用数順、被引用数順にソートすることもできる。引用数順では多くのメモ・イラストをリンクしているハブ的な役割を持つ画像が、被引用数順では多くのメモ・イラストからリンクされている汎用性の高い画像が優先的に表示される。

³<https://developer.mozilla.org/ja/docs/Web/HTML/Element/img>

⁴<https://developer.mozilla.org/ja/docs/Web/HTML/Element/iframe>

⁵<https://developer.mozilla.org/ja/docs/Web/HTML/Element/object>

第4章 実装

本章では第3章で述べたシステムの設計を受け、DrawWikiの実装について述べる。

4.1 アプリケーション構成

DrawWiki は Web アプリケーションとして実装されているため HTML5 と SVG1.1 に準拠したブラウザがあれば、OS やデバイスに依存せず利用することができる。本アプリケーションの構成は以下の図の通りである。



図 4.1: アプリケーションの構成

4.2 クライアントサイド

実際に手書きメモ・イラストの作成や関連画像の表示を行うクライアントサイドのプログラムは HTML¹と javascript²によって実装されている。開発には Javascript にコンパイル可能な漸進的型付け言語 TypeScript³を用いている。

4.2.1 手書きデータの取得

DrawWikiにおいて指やスタイラスの操作から生じる座標等の手書きデータを PointerEvent API⁴によって取得している。この API は TouchEvent や MouseEvent と異なりタッチやマウスだけでなくスタイラスを含めたあらゆるユーザー入力を透過的に扱うことが可能で(図 4.2)、また主要なブラウザに全て実装されているためあらゆるプラットフォーム・デバイスから利用できる。

4.2.2 手書きデータの描画

DrawWiki はミリ秒単位の頻度で発生する PointerEvent から動的に手書きストロークを生成し、選択された要素をグループ化した上でハイパーリンクを追加する等の SVG に対する

¹<https://developer.mozilla.org/ja/docs/Web/HTML>

²<https://developer.mozilla.org/ja/docs/Glossary/JavaScript>

³<https://www.typescriptlang.org/>

⁴<https://developer.mozilla.org/ja/docs/Web/API/PointerEvent>

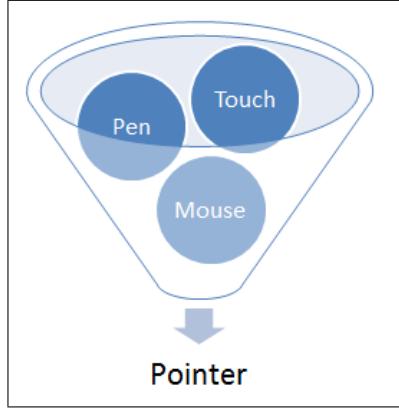


図 4.2: PointerEvent API の概要

複雑な操作を行っている。一般的に SVG の操作は HTML と同じく DOM インターフェース (appendChild や removeChild、insertBefore 等) を用いて行うが、操作の度に SVG の DOM にアクセスすることは処理の複雑化と描画パフォーマンスの低下を招く。そこで DrawWiki では実際に表示されている SVG の DOM とは別に、プログラム上で仮想的な DOM を構築し、要素の属性変更やハイパーリンクの埋め込み等の処理を仮想 DOM 上で行い、その操作が完了した段階で実際の DOM との差分を Dispatch するという手法を用いている。これによって処理の簡略化と描画パフォーマンスの向上を実現している。仮想 DOM 構築と実 DOM への反映は View ライブリの React⁵を利用している。

4.2.3 debounce された更新処理

DrawWiki は手書きメモ・イラストが編集され変更が生じる度に自動でデータを上書きし保存するが、画面上で指やスタイルスペンを動かしている間は常に PointerEvent が発生し続けているため、この頻度で変更をアップロードするとサーバー側の処理能力を超えてしまう。そこで Event 発生後にタイマーをセットし、2 秒経過した段階でアップロード処理を行う debounce 機能を実装した。このタイマーは新しい Event が発生する度にリセットされるため、ペンを置いてしばらく、つまり最後の Event から 2 秒間新たな Event が発生しないことが確定してからアップロード処理が行われる。これによりアプリケーションサーバーやアセットサーバーに過大な負荷を与えることなく更新処理を行うことができる。

4.2.4 リンク付要素の視覚的表現

関連画像ビューにはリンクしている手書きメモ・イラストのサムネイルが表示される。そのサムネイルを選択するとリンクされているキャンバス内の要素が視覚的に変化し、対応関

⁵<https://ja.reactjs.org/>

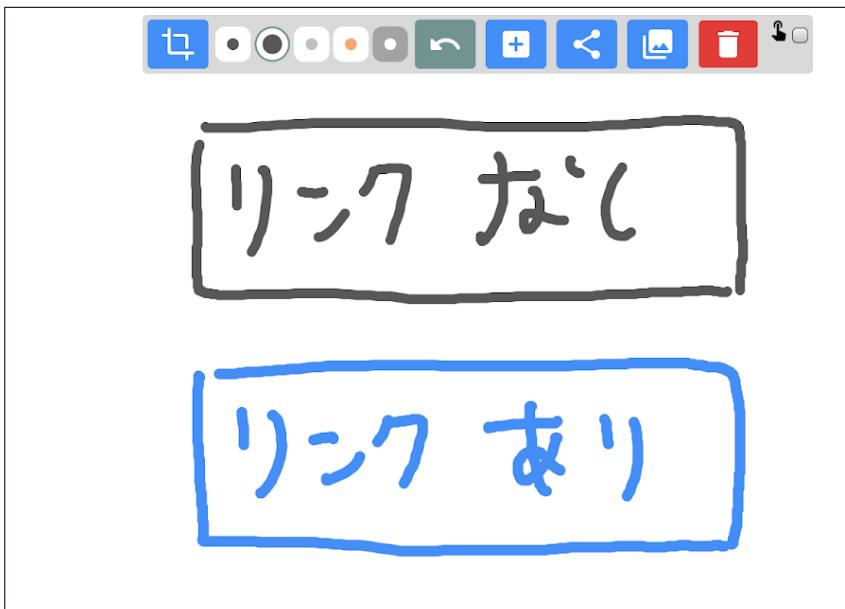


図 4.3: リンクされていない要素(上)とされている要素(下)

係が強調表示される(図 4.3)。SVG はその仕様上 CSS⁶を適用することも可能なためこの視覚効果は CSS と CSS @keyframes⁷を組み合わせることで実現している。

4.3 サーバーサイド

サーバーサイドはアプリケーションサーバーとアセットサーバーとで構成されている。

4.3.1 アプリケーションサーバー

DrawWiki は Node.js⁸上で動作する Web アプリケーションとして実装されている。HTTP リクエストを処理する Web アプリケーションフレームワークとして Express⁹を用い、そのホスティング環境として BaaS(Backend-as-a-Service) の一つである Heroku¹⁰を利用している。

(1) 手書きメモ・イラストのアップロード処理

クライアントは作成した手書きメモ・イラストを”multipart/form-data”形式でエンコーディングして送信し、アプリケーションサーバーはそのデータを POST 通信で受け取る。そ

⁶<https://developer.mozilla.org/ja/docs/Web/CSS>

⁷<https://developer.mozilla.org/ja/docs/Web/CSS/@keyframes>

⁸<https://nodejs.org/>

⁹<https://expressjs.com/>

¹⁰<https://www.heroku.com/>

の後は自動生成されるメタデータ（ソースコード 4.1）とともにデータをアセットサーバーにアップロードする。

また既存のメモ・イラストやメタデータの読み取り、更新、削除等の操作は REST 原則 [5]に基づいたルーティングによって処理される。

(2) メタデータの構成

手書きメモ・イラストの本体データとは別に、以下のようなメタ情報を管理するために DrawWiki ではメタデータファイルも取り扱う。

- メモ・イラストの作成者
- 最終更新日時
- 引用している/引用されているメモ・イラストのリスト

ソースコード 4.1: メタデータの概要

```
1  export type HyperIllust = {
2    id: string; //手書きメモ・イラストの ID
3    sourceKey: string; //本体 SVG ファイルの Key
4    sourceURL: string; //本体 SVG ファイルの URL(フルパス)
5    size?: number; //ファイルサイズ
6    linkedList?: string[]; //リンクされているメモ・イラストの ID リスト
7    linkedByList?: string[]; //この画像をリンクしているメモ・イラストの ID リスト
8    createdAt: DateLike; //作成日時
9    updatedAt: DateLike; //更新日時
10   owner: string; //作成者の名前
11};
```

4.3.2 アセットサーバー

アプリケーションサーバーはステートレスなプロセスとして実行されるため、変数やファイル等を保存し永続化する仕組みをもたない。ファイルとしてのメモ・イラストやメタデータを保存するため DrawWiki はアセットサーバーとしてクラウドストレージである AWS S3¹¹を利用している。アップロードされたメモ・イラストは DrawWiki 以外の Web サイトからも閲覧・再利用できるように、ACL(Access-Control-List) を公開読み取り（"public-read"）に設定している。またアセットサーバー自体もオリジン間リソース共有¹² を有効化している。

¹¹<https://aws.amazon.com/jp/s3/>

¹²<https://developer.mozilla.org/ja/docs/Web/HTTP/CORS>

第5章 利用例

本章では DrawWiki の利用例を紹介する。

手書きメモ・イラストの内部に別の画像へのリンクが埋め込めるという DrawWiki の機能を活かし、次のように利用した。

5.1 グラフィカルなアイデasketchやメモ

5.2 スケッチによるソフトウェアのモックアップの作成

ソフトウェアの画面や挙動を設計する際に、柔軟な表現が可能な手書きスケッチを用いることが有効なデザインプラクティスとして知られている。DrawWiki を開発する際も、機能や画面全体のデザインを考える際に DrawWiki を用いてスケッチを行いながら検討した。リンクが埋め込まれた要素をクリックするとリンク先の画像が表示されることから、これをインタラクションや画面遷移に見立てることにより図??のようなナビゲーションを伴ったソフトウェアのモックアップを手書きスケッチによって作成することができる。

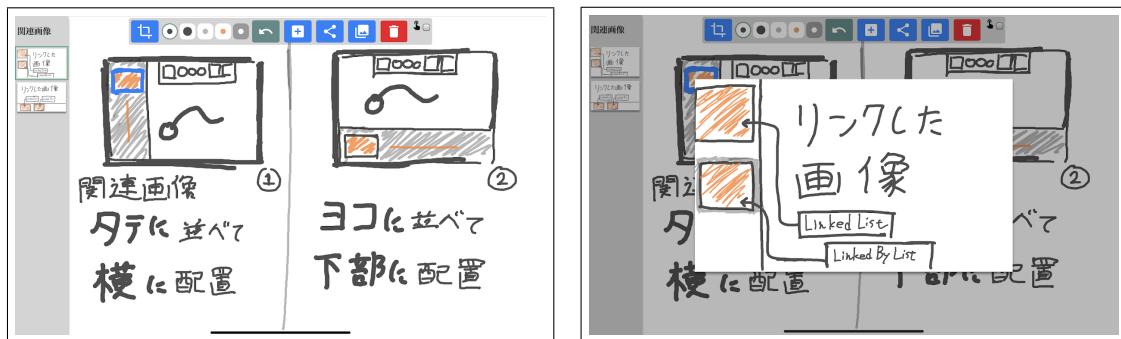


図 5.1: DrawWik を用いて作成された Drawwiki のモックアップ

図 5.2: アクションによってポップアップするダイアログの再現

5.3 インタラクティブな図解

自動車等の整備を行う際は、製品マニュアルを熟読することで構造を理解し、交換すべき部品をパーツリストで確認し、さらに取り付けに必要なワッシャ・ボルトのサイズやそれらの適正な締め付けトルクを念頭に置いた上で行う必要があるが、それらの情報は異なるページや冊子に分かれて記載されていることが一般的であり、参照が大変である。(図 5.3, 図 5.4)

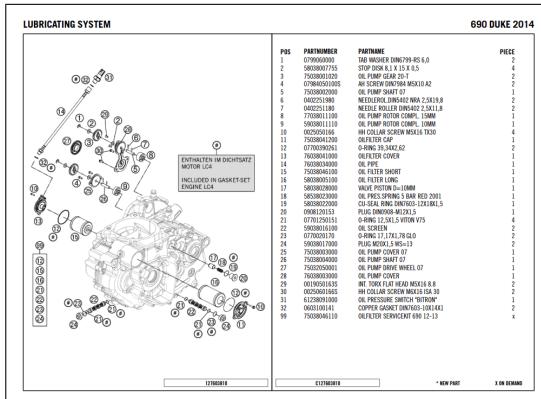


図 5.3: 既存のパーツリスト

| 22 TECHNICAL DATA | | | |
|--|-------|---------------------------|--------------|
| Screw, oil filter cover | M5x16 | 6 Nm (4.4 lbf ft) | - |
| Screw, oil pump cover | M6 | 5 Nm (3.4 lbf ft) | Locite® 243™ |
| Screw, oil pump cover, top | M5 | 6 Nm (4.4 lbf ft) | Locite® 243™ |
| Chain securing guide | M6 | 5 Nm (3.7 lbf ft) | - |
| Cylinder head screw | M6x25 | 10 Nm (7.4 lbf ft) | Locite® 243™ |
| Plug, vacuum connection | M6 | 10 Nm (7.4 lbf ft) | Locite® 243™ |
| Remaining screws, engine | M6 | 10 Nm (7.4 lbf ft) | - |
| Screw, alternator cover | M6x25 | 10 Nm (7.4 lbf ft) | - |
| Screw, alternator cover (chain shaft through-hole) | M6x25 | 10 Nm (7.4 lbf ft) | Locite® 243™ |
| Screw, autocompression | M6 | 3...4 Nm (2.2...3 lbf ft) | Locite® 243™ |
| Screw, axial lock of camshaft | M6 | 10 Nm (7.4 lbf ft) | Locite® 243™ |
| Screw, clutch cover | M6 | 10 Nm (7.4 lbf ft) | - |
| Screw, clutch slave cylinder | M6x20 | 10 Nm (7.4 lbf ft) | Locite® 243™ |
| Screw, clutch slave cylinder | M6x35 | 10 Nm (7.4 lbf ft) | - |
| Screw, cylinder position sensor | M6x16 | 10 Nm (7.4 lbf ft) | Locite® 243™ |
| Screw, cylinder | M6 | 10 Nm (7.4 lbf ft) | Locite® 243™ |
| Screw, engine case | M6 | 10 Nm (7.4 lbf ft) | - |
| Screw, ignition coil | M6 | 10 Nm (7.4 lbf ft) | - |
| Screw, locking lever | M6x20 | 10 Nm (7.4 lbf ft) | Locite® 243™ |
| Screw, oil pump cover, bottom | M6 | 10 Nm (7.4 lbf ft) | Locite® 243™ |
| Screw, rocker arm shaft | M6x30 | 12 Nm (8.9 lbf ft) | - |
| Screw, shift drum locating | M6x30 | 10 Nm (7.4 lbf ft) | Locite® 243™ |
| Screw, shift lever | M6 | 14 Nm (10.3 lbf ft) | Locite® 243™ |
| Screw, starter motor | M6x20 | 10 Nm (7.4 lbf ft) | Locite® 243™ |

図 5.4: 既存の部品情報リスト

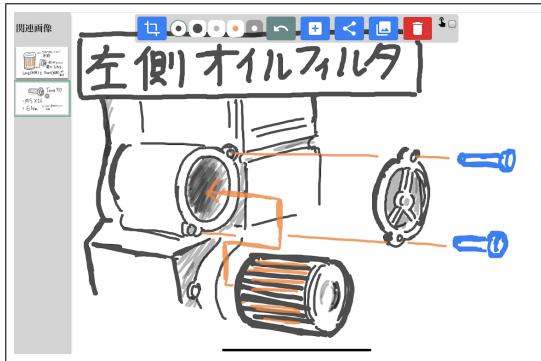


図 5.5: DrawWiki による自作整備メモ



図 5.6: 関連情報を表示した画面

DrawWiki では製品の構造を理解しやすいよう自由なレイアウトで簡潔に書いた図に部品や組み立てに要する注意等のリストが描かれた別のメモをリンクさせることで、必要な情報へ素早くアクセスする手段を確保しながら、画面が注釈で埋まることのない視認性に優れた図解を作成することが可能である。またリンク構造によって同じ部品や工具を用いる他のメモも推薦されるため、整備の手順を考える手助けとなる効果もある。

5.4 リンクに基づいたフラットな手書きメモの管理

これらのメモは全く異なる分野を取り扱ったものであり、従来のメモアプリケーションでは異なるフォルダに格納する、またはタグやラベル等を貼り付ける等の手段で分類されるべきものだったが、ハイパーイラスト内のリンク情報に基づいて関連イラストが表示される DrawWiki では、情報の整理や適切な管理を特別心かけなくともリンクを辿ることで目的のハイパーイラストを参照することが可能である。

5.5 ナレッジの共有と共同編集

DrawWiki によって作成されたハイパーイラストには URL が割り振られているため、画像として他の Web サイト内に埋めこむことも可能である。この状態でもリンク情報は失われずファイル内に内包されているため、リンクされている関連画像も含めたナレッジとして共有することができる。さらに他のユーザーによる内容の追記や、別のハイパーイラストをリンクさせるといった操作にも対応しているため手書きメモを取るような気軽さで Wiki のようなコラボレーションを行うことができる。

5.6 まとめ

本章では、本システムによって実現可能な利用例について述べた。ハイパーイラストと手書きベース Wiki の組み合わせによって既存の手書きメモ・イラストの問題点を解決でき、またテキストベース Wiki に対しても優れた点があることがわかった。本章で述べた応用例に限らず、様々な応用が可能と考えられる。

第6章 関連研究

本節では、手書きベース Wiki のコンセプトや機能に関する興味深い研究を紹介し、それらの特徴を示しつつ本研究との比較を行う。

6.1 手書きデータをハイパーテキストとして扱うシステム

手書きのメモやイラスト等の画像データに対してハイパーリンクを埋めこむ機能を持つものとして以下のようなシステムが挙げられる。

6.1.1 HyperCard

Appleによって開発されたHyperCard¹は、マルチメディアやリンクを埋め込んだハイパーテキストのようなコンテンツを作成できるソフトウェアである。HyperCardのコンテンツは複数のカードから構成されたスタックという形式で運用されるが、図6.1のように手書きの図の上にフィールドを作成し、マウスでクリックした際に異なるカードへの移動するという処理をHyperTalkと呼ばれるスクリプト言語で記述することでハイパーリンクに相当する機能を実現することができる。

DrawWikiとの比較 HyperCardではあらゆる手書きの図を含むあらゆるオブジェクトに対してカード間リンクを埋めこむことができるがその設定にはコーディングが必要なため敷居が高い。DrawWikiは範囲選択ツールで要素を選び、リンク先のメモ・イラストを設定するだけで同じ操作ができるため手数が少なく、より手軽であると言える。またカード間は自在に移動できるが異なるマシンに置かれたスタックに対してアクセスすることができない。DrawWikiでは全てのメモ・イラストがURLを持ったハイパーテキストであるため全世界からアクセス可能である。

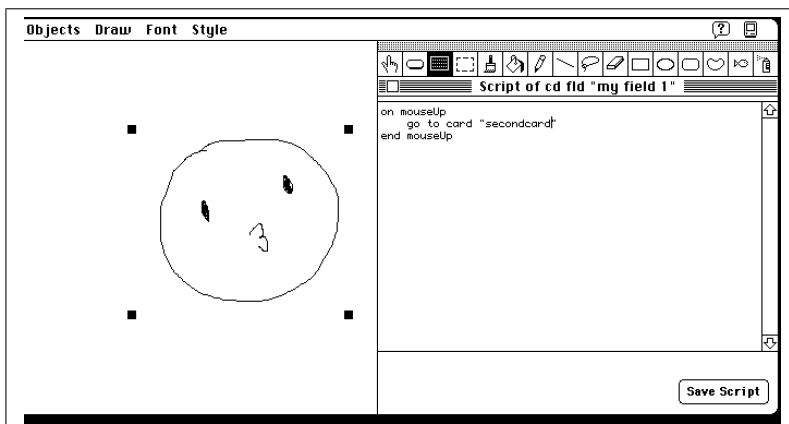


図 6.1: エミュレータ上で再現された HyperCard

¹<http://www.apple.com/hypercard>

6.1.2 Adobe Photoshop

Adobe Systems²が開発する画像編集ソフトウェアである Photoshop³には作成した画像を HTML ファイルとして出力するオプションが存在する。設定によっては任意の領域に対してハイパーリンクを埋めこむことができる。この時出力されるファイルは、本体となる画像とそれをオーバーレイする HTML ファイルとセットで構成される。

DrawWiki との比較 そもそも Photoshop はプロフェッショナル向けのツールであり、すばやく手書きメモを取るという用途で使うには大袈裟すぎる。また画像付き HTML ファイルとして出力した場合双方を正しい位置関係に配置したディレクトリごと配布されなければならないため、1 ファイルで画像もハイパーリンクも記述できる SVG と比較してポータビリティに欠ける。

6.1.3 Inkscape

The Inkscape Team⁴によって開発されるオープンソースのベクター画像編集ソフトウェアである Inkscape⁵は画像として表現されている SVG をテキストエディタによって直接編集する機能を備える。これにより複数の要素をグループ化し、同一のハイパーリンクを埋めこむ等の柔軟な操作を行うことができる。

DrawWiki との比較 Photoshop と同様 Inkscape も手書きメモという用途に対して高機能すぎる。また SVG を直接編集するには SVG や XML の仕様についてよく理解している必要があるため敷居が高い。DrawWiki では SVG の構造をユーザーが意識することなく同一の操作を実現できる。また Photoshop と同様サポートしている機能は画像に対するハイパーリンクの埋め込みのみで、ホスティング等は行わない。したがって作った画像を参照可能にするためにはユーザー自らがサーバーにアップロードしなければならないため、Wiki ツールとして活用するには機能不足であると言える。

6.2 手書きデータによって文書の参照や管理・再利用を支援するシステム

メモやイラストに限らず文字を書く、下線を引く、丸で囲む等の手書きデータを文書の閲覧や検索、再利用に活用する試みとして以下のシステムが挙げられる。

²<https://www.adobe.com/>

³<https://www.adobe.com/jp/products/photoshop.html>

⁴<https://inkscape.org/user/teams/>

⁵<https://inkscape.org/>

6.2.1 XLibris

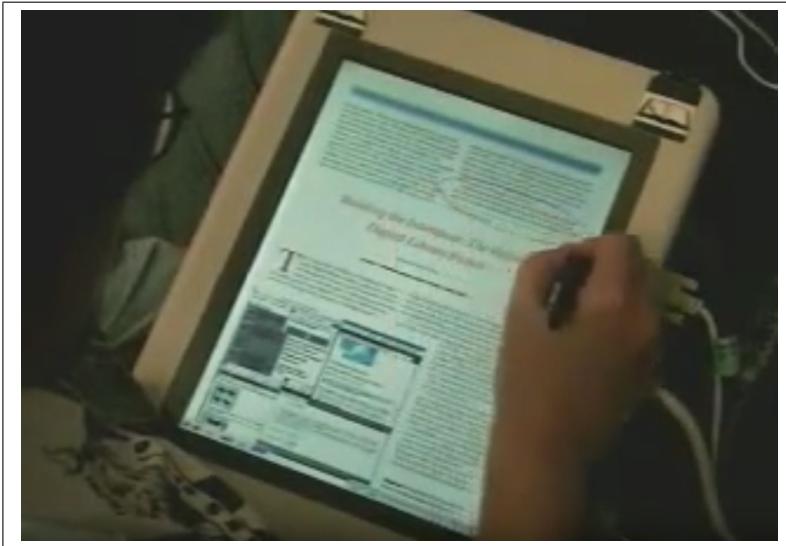


図 6.2: 操作中の XLibris

Price らによる XLibris[13](図 6.2) はタブレットデバイス上で Active Reading を行うことの目的としたシステムである。スタイルスペンによって手書きの注釈やハイライトを文章中に自在に追記できるほか、追記した部分をリンクとしてページの参照に用いたり、ハイライトされた単語や文章を検索に利用し関連資料を余白に表示する機能を備えている。

DrawWiki との比較 メモやイラストではなく文書に追記する注釈を主眼に置いているというコンセプトの相違があるものの、手書きの注釈がリンクとなり文書を参照しやすくなる機能や、追記されたページを関連資料として表示する機能が Active Reading といった用途においても有用であることが示されている。

6.2.2 InkSeine

Hinckley らによって開発された InkSeine[6](図 6.3) はペンインターフェースを備えたタブレット PC における利用を想定した電子スクラップブックシステムである。キーボードは使わずにペンのみで操作が完結するよう UI が設計されており、テキスト入力も手書き文字認識によって行う。また入力した手書き文字を Web ページやローカルファイルの検索に活用したり、検索結果のスクリーンショットをリンクと共にキャンバスに貼り付ける等の機能を備えており、積極的な情報の再利用を実現している。

DrawWiki との比較 手書きの文字を認識し、検索結果をリンクと共にノートに配置するという機能によってペンベースの個人用電子スクラップブックとして活用することは可能であるものの、作成したノートはファイルとして保存されるため、URL 等から参照したり共同編集することができず、Wiki システムとして活用することは不可能である。

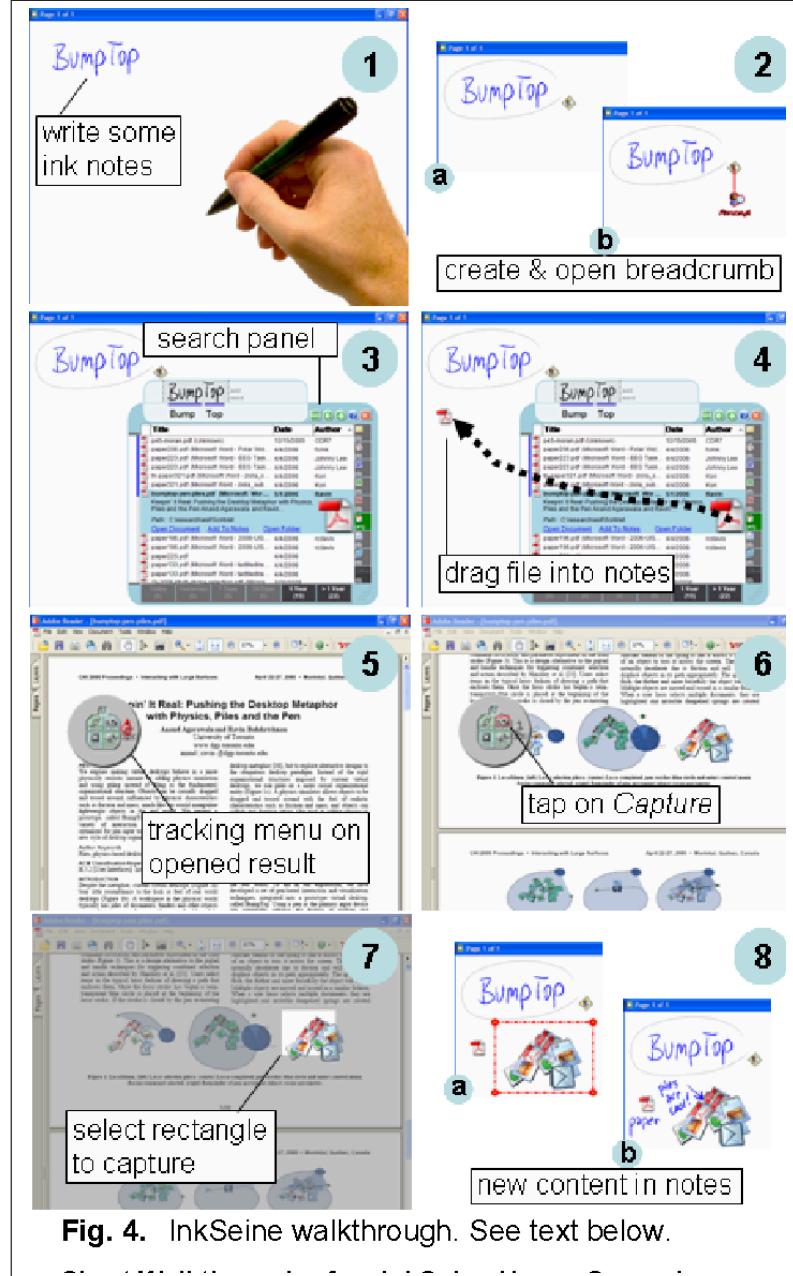


図 6.3: InkSeine の操作画面

6.3 手書きスケッチをプロトタイピングに応用するシステム

アプリケーションや Web サイトの UI をデザインするごく初期の段階ではスケッチがよく利用される [11]。スケッチとしての手書きデータをプロトタイピングに取り入れる試みとして以下のシステムが挙げられる。

6.3.1 DENIM

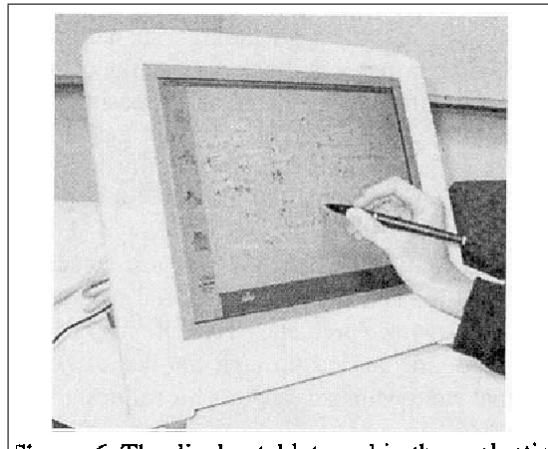
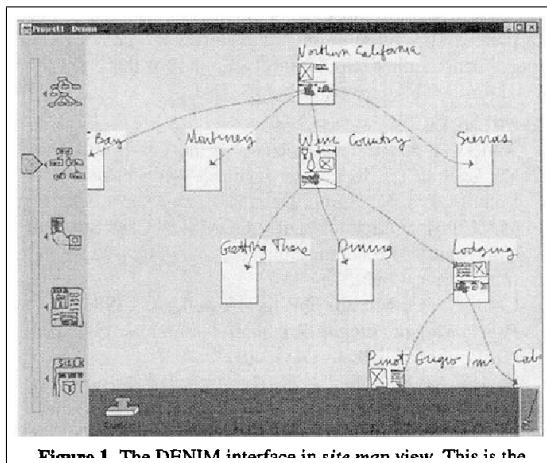


図 6.4: DENIM の画面

図 6.5: 操作を行うタブレット機器

Lin はタブレットからのスケッチを元に Web サイトのデザインやナビゲーションの設計の支援を行うシステム DENIM を開発した [10](図 6.4)。図 6.5 のようなタブレット機器から操作するよう設計されており、Web ページを手書きでスケッチし、それらを線で結ぶとリンクによるナビゲーションを再現した Web サイトのモックアップを作成することができる。また Web ページの細部のデザインといった小さな視点から、複数のページ間の繋がりを俯瞰的に確認するという大きな視点までを柔軟にカバーするために、スケッチ用のキャンバスの横に視点レベルを変更可能なレンジバーが備えられており、ズーミングインターフェースの知見が取り入れられているのが特徴である。

6.3.2 SILK

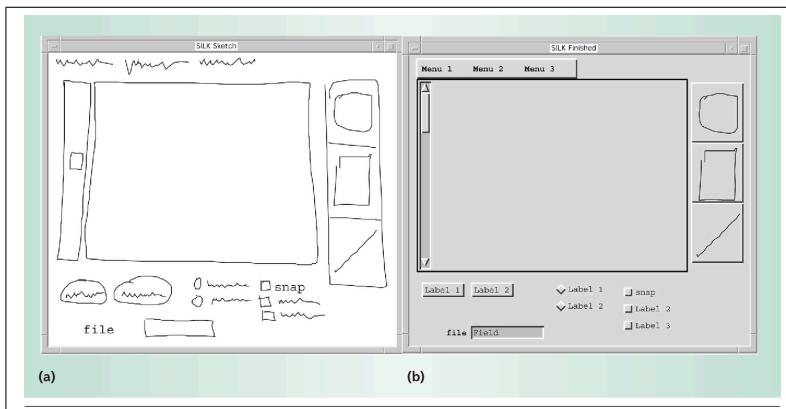


図 6.6: SILK の画面

Landay らはスタイルスから入力した手書きデータから実際に動作する GUI を構築する SILK (Sketching Interfaces Like Krazy) というシステムを実装した [7](図 6.6)。このシステムは手書きデータの形状からボタンやフォーム等の UI コンポーネントを類推し、配置することで動作可能なソフトウェアのプロトタイプを構築することができる。

DrawWiki との比較 いずれのシステム(第 6.3.1 節、第 6.3.2 節)も、ラフスケッチの段階である程度のインタラクションを再現するプロトタイプを作成可能なことがデザインを行う上で役立つことを示している。DrawWiki はデザインを専門とするツールではないが、手書きスケッチの内部へのリンク埋め込みや、リンク先画像のダイアログ表示といった機能を用いることで第 5.2 節のように「インタラクションを伴うプロトタイプ」として応用することができる。したがって DrawWiki はデザインの分野においても応用可能なツールであると言える。

6.4 手書きスケッチの中の要素に注釈を埋め込めるシステム

手書きのスケッチの内部に、手書きのメモ・イラストや画像、音声等のメディアを注釈として埋めこむシステムとして以下が挙げられる。

6.4.1 SketchComm



Figure 1. SketchComm being used by a designer.

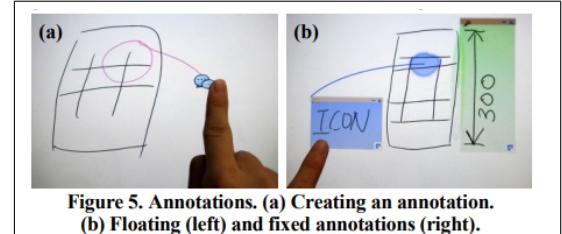


Figure 5. Annotations. (a) Creating an annotation.
(b) Floating (left) and fixed annotations (right).

図 6.8:

図 6.7:

Li らによる SketchComm[9] はアイデアに関するスケッチが非同期的にレビューされることを念頭に置いて設計されたシステムである(図 6.7)。対面によるコミュニケーションを伴う同期的なレビューと異なり、非同期的なレビューではコンテキストが欠落する恐れがあるため、それを補うために注釈やコメント、音声データ等をスケッチの内部に埋め込むことができる(図 6.8)。これらのコンテンツはキャンバスを専有しないよう必要に応じて表示/非表示を切り替えられる。このシステムを用いた評価実験を行ったところ、「注釈を活用することでスケッチの視認性を確保しながらも必要な情報は適宜参照できること」が実験に参加し

たデザイナーから高く評価されていた。また OneNote⁶との比較も行われており、「同じコンセプトを表現する場合、注釈が利用できない OneNote では多くの情報をキャンバスに描かなければならぬためスケッチが見づらくなつた」「SketchComm の方が OneNote よりも少ない労力で多くを表現できる」という評価がなされている。

DrawWiki との比較 DrawWiki が手書きメモ・イラストの中に埋めこんむのはハイパリンク (URL) であるが、クリックすると遷移するのではなくリンク先の画像がダイアログで表示されるため、第 5.2 節や第 5.3 節のように「表示/非表示を切り替え可能な注釈」として活用することができる。スケッチの内部に関連する情報をあわせて保存できることがコンセプトの理解を助けることをこの研究は示しており、この機能は Wiki システムにおいても有効であると考えられる。

⁶<https://www.onenote.com/>

第7章 考察

本章では、手書きベース Wiki の利用者の意見や自身の運用経験やをまとめシステムの有用性を評価し、諸問題や研究の重要性・発展性について述べる。

7.1 評価

本研究によって実装した手書きベース Wiki を、以下の項目について評価する。

1. 第 2.3 節で述べた手書きメモ・イラストの問題点を解決できたか
 - 参照や管理が面倒
 - 再利用が難しい
2. 手書きメモ・イラストがハイパーリンクを持つことの有用性
3. 手書きメモ・イラストが Wiki 的に扱えることの有用性

手書きベース Wiki のプロトタイプとなる「DrawWiki」は 2019 年 10 月に Web アプリケーションとして公開した¹。また ORF2019²にて DrawWiki の展示発表を行った。本節では、DrawWiki の展示発表で得られた感想、ユーザーからのフィードバックや筆者の運用経験を元にシステムを評価する。

7.1.1 展示発表

手書きベース Wiki のプロトタイプとなる「DrawWiki」は 2019 年 10 月に Web アプリケーションとして公開した³。また ORF2019⁴にて DrawWiki の展示発表を行った。

7.1.2 意見・感想

第 2.3 節で述べた手書きメモ・イラストの問題点について多くの同意が得られた。多くのユーザーが同様の問題を抱えつつも、本質的に解決する手段・方法が無かったため各々が運用上の工夫・苦労を強いられていた。その他に以下のような感想や意見が寄せられた。

関連資料のナレッジ化 デザインの課題では多数のスケッチを描く場面があるが、それらは紙で管理されているため管理しなければ散逸してしまい、また振り返りのために参照するのも大変である。これを既存のメモアプリケーションに置き換えても紙と同様の参照・管理の不便さが解決しないが、スケッチにリンクを追加することで関連する作品を漏れなく参照することができ、生きたナレッジとして活用できるのでは無いかという意見を得た。

¹<https://draw-wiki.herokuapp.com/>

²<https://orf.sfc.keio.ac.jp/2019/>

³<https://draw-wiki.herokuapp.com/>

⁴<https://orf.sfc.keio.ac.jp/2019/>

手書きで作成できる Wiki タイピングや PC の操作が得意でないためにテキストベース Wiki を使う機会がなかった人からも、手書きのメモを描いてそれが繋がるのであれば自分でもコンテンツを作れそうという意見があった。第??節で示したように、手軽にハイパーイラストが作成でき、それらをコンテンツとする Wiki があれば、高度なスキルがなくてもカジュアルに Wiki に参加することが可能であると考えられる。

7.1.3 筆者の運用経験

主に個人用のメモとして活用しつつ、DrawWiki の開発と並行して 3ヶ月間利用した。

ハイパーイラストの有用性 第??章で示したように、ソフトウェアのモックアップや自作の整備メモ等の従来のメモアプリでは表現できなかったハイパーイラストならではのコンテンツを作成することができた。ハイパーイラストの作成そのものは Inkscape⁵等の SVG を編集できるツールを用いれば可能であるが、

- 手書きの画像データを用意し
- 紐付けたい他の画像データをクラウドストレージにアップロードし
- その URL を埋め込んだ上で SVG として保存する

という一連の作業を全て手作業で行う必要があり、気軽に作成することはできなかった。DrawWiki によって簡単な操作でイラストの内部へのリンクの追加が可能になったため、手書きメモというカジュアルな用途でもハイパーイラストを作成し、その恩恵を受けられるようになった。

リンクに基づく優れた参照性 DrawWiki で作成したメモ・イラストにはリンクこそ埋め込まれているものの、フォルダやタグ・ラベル等のメモを分類し管理するための機能は存在しない。全てのメモ・イラストがフラットに置かれているものの、リンク情報に基づいて関連するメモ・イラストが推薦されるため、参照に困ることはなかった。既存のメモアプリではメモを描く前に

- どのようなファイル名にするか?
- どのようなフォルダに分類するのか?
- どのようなタグ・ラベルをつけるのか?

等を意識する必要があったが、そのような管理の工夫をせずとも、リンクによるシンプルなナビゲーションがあれば目的のメモ・イラストを参照できることができた。

7.1.4 問題点・要望

以下のような問題点が明らかになった。

⁵<https://inkscape.org/>

1. テキスト検索によってメモを参照したい

メモの内部にある手書きの文字を対象に、テキスト検索によって参照できる機能がほしいという要望があった。現状の DrawWiki において文字として描かれた手書きストロークを特別に区別・認識していないため、テキストによって検索する機能を備えていない。手書きメモであってもテキストによって全文検索したい

2. 編集履歴が分かりづらい

基本的にハイパーイラストは URL を通じていつでも、誰でも編集可能であるが、どのように編集されたのか、どのような内容が追記されたのかを判断しづらい場合がある。また手書きストロークが誰によって追加されたのかを視覚的に表示する機能を備えていないため、共同編集を行った際に誰による編集なのかを把握しづらいという指摘があった。

3. リンクを追加する際の補助が欲しい

他のメモへのハイパーリンクを埋め込む際に、どのメモを埋め込むかはモーダルによって選ぶ DrawWiki ではある要素に対して別のメモ・イラストをリンクさせたる際に、今までに作成したメモ・イラストの一覧から選択するという操作になっている。この一覧は時系列順や引用数/被引用数によってソート・絞り込みを行うことが可能だが、さらに進んでメモを取っている最中にその時点での手書きストロークに類似する過去のメモやイラストを候補として推薦する仕組みがあれば、よりスムーズなリンクの埋めこみやイラストのインポートが可能になるという意見があった。

7.2 考察

7.2.1 設計指針の妥当性

本システムは既存のメモアプリケーションとは異なる利用形態を持つが、実際に利用した、またはデモを体験したユーザーからは概ね好意的に受け止められたため、ハイパーリンクと Wiki の組み合わせという設計は適切であったと言える。本システムをベースに様々な改良を重ねることで、より優れた手書きメモ・イラストの利用環境の実現や応用が可能であると考えられる。

7.2.2 解決すべき課題

1. テキスト検索による参照

DrawWiki ではスタイラスから得られる手書きデータを元にストロークを生成しているため、オンライン手書き文字認識を応用することでメモ内のテキストを認識し、 クエリとして検索・参照する機能は技術的に実装可能である。テキスト検索機能を備えることで、よりメモを参照しやすい状態で運用することができると考えられる。

2. 編集履歴の可視化

DrawWiki で管理されるハイパーイラストでは編集に関する履歴情報を保持している。これらの情報を元に、編集者によってストロークの色や形状に変更を加えたり、新しく編集された部分について強調表示したり等の視覚表現を追加する機能があればより便利に共同編集を行うことが可能になると考えられる。

3. ”手書き入力補完”によるリンク追加処理の支援

スケッチしている図形を認識したり [12][14]、それをクエリとして画像検索を行う手法は数多く存在する [3][4] [1]。これらを応用することで描画中の手書きストロークを元に過去に作成したメモ・イラストの中から類似したものを検索し、候補として提案する”手書き入力補完”のような機能を実装することができ、よりスムーズに手書きメモ・イラスト同士をリンクさせることができるようになる。

7.2.3 手書きメモ・イラストの問題点の検証

本システムにおいて第 2.3 節で述べた問題点が克服されているかどうかを検証する。

- 参照や管理が面倒

手書きメモ・イラストのリンク情報に基づいて関連イラストを推薦する機能により、管理に労力を割くことなく目的のメモ・イラストを参照できるようになった。

- 再利用が難しい

手書きメモやイラスト同士をリンクさせたり、他の Web サイトに埋め込んだりといった再利用が可能になった。

以上のように、第 2.3 節で述べた問題点がハイパーイラストと手書きベース Wiki によって解決した。

第8章 結論

本章では本研究を総括する。

8.1 研究の成果

本研究では、ハイパーリンクを内蔵した次世代の手書きデータ記述形式「ハイパーイラスト」と、それをコンテンツとして扱う Wiki システム「手書きベース Wiki」の提案を行った。

まず第 2 章において、既存の手書きメモ・イラストの問題点をテキストの進化と比較しながら分析した。計算機上で手書きメモ・イラストを扱う既存システムの現状をとりあげ、計算機が普及した現在も根本的に解決されていないことを示した。

第 3 章では、第 2 章で述べた手書きメモの問題点に対する解決方法として「手書きベース Wiki」を提案した。また、「手書きベース Wiki」のプロトタイプとして実装した「DrawWiki」の機能を使い方と共に解説した。

第 4 章では、「DrawWiki」のアプリケーション構成と詳細な実装について述べた。

第 5 章では、「DrawWiki」によって実現可能な利用例を紹介した。

第 6 章では、本研究に関連する研究を紹介し、それぞれのアプローチの特徴と問題点を分析し、本研究との比較検討を行った。

第 7 章では、ユーザーからのフィードバックや筆者による運用経験をもとに本研究の有効性と問題点を分析した。

8.2 総括

本研究では手書きのメモやイラストの中に自在にハイパーリンクを埋め込み、それらをナレッジとして扱える「手書きベース Wiki」の開発を行った。「手書きベース Wiki」はハイパーリンクや Wiki 等の技術の組み合わせによって、参照や管理、再利用が難しいといった手書きのメモ・イラスト問題点を解決した。また新しい活用法を実現した。さらにテキストベース Wiki に対しても優れた点があることを示した。今後は第 7 章で述べた問題点を受け、システムを改善していく。

謝辞

慶應義塾大学環境情報学部 増井俊之教授には学部から5年間の長きに渡りご指導を賜りました。深く感謝いたします。また、本研究の副査としてご意見、ご助言を頂きました中西泰人教授、武田圭史教授に感謝いたします。また自身の研究について幅広い議論をしていたいた政策・メディア研究科博士課程の田中優氏、大和比呂志氏を初め、様々な形でアドバイスをくださった増井俊之研究会OB諸氏に感謝いたします。

2020年1月 慶應義塾大学 政策・メディア研究科 修士2年 早川匠

参考文献

- [1] Tao Chen, Ming-Ming Cheng, Ping Tan, Ariel Shamir, and Shi-Min Hu. Sketch2photo: internet image montage. In *SIGGRAPH 2009*, 2009.
- [2] Dean Jackson Chris Lilley. Scalable vector graphics (svg). 10 2004.
- [3] Mathias Eitz, Ronald Richter, Tamy Boubekeur, Kristian Hildebrand, and Marc Alexa. Sketch-based shape retrieval. *ACM Trans. Graph.*, Vol. 31, pp. 31:1–31:10, 2012.
- [4] Mathias Eitz, Ronald Richter, Kristian Hildebrand, Tamy Boubekeur, and Marc Alexa. Photosketcher: Interactive sketch-based image synthesis. *IEEE Computer Graphics and Applications*, Vol. 31, pp. 56–66, 2011.
- [5] Roy T. Fielding. Architectural styles and the design of network-based software architectures (chapter 5). 2000.
- [6] Ken Hinckley, Shengdong Zhao, Raman Sarin, Patrick Baudisch, Edward Cutrell, Michael Shilman, and Desney S. Tan. Inkseine: In situ search for active note taking. In *CHI*, 2007.
- [7] James A. Landay and Brad A. Myers. Sketching interfaces: Toward more human interface design. *IEEE Computer*, Vol. 34, pp. 56–64, 2001.
- [8] Bo Leuf and Ward Cunningham. The wiki way: Quick collaboration on the web. 2001.
- [9] Guang Li, Xiang Cao, Sergio Paolantonio, and Feng Tian. Sketchcomm: a tool to support rich and flexible asynchronous communication of early design ideas. In *CSCW '12*, 2012.
- [10] James Lin, Mark W. Newman, Jason I. Hong, and James A. Landay. Denim: finding a tighter fit between tools and practice for web site design. In *CHI '00*, 2000.
- [11] Mark W. Newman and James A. Landay. Sitemaps, storyboards, and specifications: a sketch of web site design practice. In *DIS '00*, 2000.
- [12] Matthew J. Notowidigdo and Robert William Clinton Miller. Off-line sketch interpretation. In *AAAI 2004*, 2004.

- [13] Morgan N. Price, Bill N. Schilit, and Gene Golovchinsky. Xlibris: the active reading machine. In *CHI '98*, 1998.
- [14] Jacob O. Wobbrock, Andrew D. Wilson, and Yang D. Li. Gestures without libraries, toolkits or training: a \$1 recognizer for user interface prototypes. In *UIST '07*, 2007.
- [15] 淳一金箱, 直蛭田, 克彦原田, 俊介高尾, 裕行佐竹, ギブソンジェームズ, 亨赤羽. 相互作用を喚起するアイデアスケッチ手法 : Interactive sketch の提案. 日本デザイン学会研究発表大会概要集, Vol. 58, pp. 14–14, 2011.
- [16] 増井俊之. Gyazz-柔軟で強力な万人のための wiki システム. 第 52 回プログラミングシンポジウム予稿集, 第 2011 卷, pp. 43–50, jan 2011.