Detection and Localization of HTML Presentation Failures Using Computer Vision-Based Techniques

Sonal Mahajan and William G. J. Halfond

Department of Computer Science
University of Southern California
Presentation of a Website

• What do we mean by presentation?
  – “Look and feel” of the website in a browser

End user – no penalty to move to another website
Business – loses out on valuable customers

  – It takes users only 50 ms to form opinion about your website (Google research – 2012)
  – Affects impressions of trustworthiness, usability, company branding, and perceived quality
Motivation

- Manual detection is difficult
  - Complex interaction between HTML, CSS, and Javascript
  - Hundreds of HTML elements + CSS properties
  - Labor intensive and error-prone

- **Our approach** – Automate debugging of presentation failures
Two Key Insights

1. Detect presentation failures

Use computer vision techniques
Two Key Insights

2. Localize to faulty HTML elements

Use rendering maps
Limitations of Existing Techniques

• **Regression Debugging**
  – Current version of the web app is modified
    • Correct bug
    • Refactor HTML (e.g., convert `<table>` layout to `<div>` layout)
  – DOM comparison techniques (XBT) not useful, if DOM has changed significantly

• **Mockup-driven development**
  – Front-end developers convert high-fidelity mockups to HTML pages
  – DOM comparison techniques cannot be used, since there is no existing DOM
  – Invariants specification techniques (Selenium, Cucumber, Sikuli) not practical, since all correctness properties need to be specified
  – Fighting layout bugs: app independent correctness checker
Running Example

Web page rendering $\neq$ Expected appearance (oracle)
Our Approach

**Goal** – Automatically detect and localize presentation failures in web pages

**P1. Detection**

**P2. Localization**
P1. Detection

• Find visual differences (presentation failures)
• Compare oracle image and test page screenshot
• **Simple approach**: strict pixel-to-pixel equivalence comparison
  – Drawbacks
    • Spurious differences due to difference in platform
    • Small differences may be “OK”
Perceptual Image Differencing (PID)

• Uses models of the human visual system
  – Spatial sensitivity
  – Luminance sensitivity

  **Shows only human perceptible differences**

  – $\Delta$ : Threshold value for perceptible difference
  – $F$ : Field of view of the observer
  – $L$ : Brightness of the display
  – $C$ : Sensitivity to colors
P1. Detection – Example

Test with: Visual comparison using PID

Filter differences belonging to dynamic areas

Oracle

Visual comparison using PID

Apply clustering (DBSCAN)

A

B

C

Sign in

Cellphone advertisement

News box

Advertisement box

Username: ____________________________

Password: ____________________________

User: ____________________________

This is indeed a big news! This is indeed a big news! This is indeed a big news! This is indeed a big news! This is indeed a big news! This is indeed a big news! This is indeed a big news! This is indeed a big news! This is indeed a big news! This is indeed a big news!
P2. Localization

- Identify the faulty HTML element

  *Use rendering maps to find faulty HTML elements corresponding to visual differences*

- Use R-tree to map pixel visual differences to HTML elements

- “R”ectangle-tree: height-balanced tree, popular to store multidimensional data
P2. Localization - Example

```html
table
<table>
<thead>
<tr>
<th>tr[2]</th>
</tr>
</thead>
<tbody>
<tr>
<td>td</td>
</tr>
<tr>
<td>Login:</td>
</tr>
<tr>
<td>table</td>
</tr>
<tr>
<td>tr</td>
</tr>
<tr>
<td>td</td>
</tr>
<tr>
<td>Username:</td>
</tr>
<tr>
<td>input</td>
</tr>
<tr>
<td>tr[2]</td>
</tr>
<tr>
<td>td</td>
</tr>
<tr>
<td>Password:</td>
</tr>
<tr>
<td>input</td>
</tr>
<tr>
<td>tr[3]</td>
</tr>
<tr>
<td>td</td>
</tr>
<tr>
<td>input</td>
</tr>
<tr>
<td>Sign in</td>
</tr>
</tbody>
</table>
```
P2. Localization - Example

Sub-tree of R-tree
P2. Localization - Example

Result Set:

/html/body/…/tr[2]
/html/body/…/tr[2]/td[1]
/html/body/…/tr[2]/td[1]/table[1]
/html/body/…/tr[2]/td[1]/table[1]/tr[1]
/html/body/…/tr[2]/td[1]/table[1]/td[1]

Map pixel visual differences to HTML elements
Special Regions Handling

- Special regions = Dynamic portions (actual content not known)

1. Exclusion Region
2. Dynamic Text Region
1. Exclusion Regions

- Only apply size bounding property
2. Dynamic Text Regions

• Style properties of text known

Test web page

Modified test web page (Oracle)

Run P1, P2
P3. Result Set Processing

- Rank the HTML elements in the order of likelihood of being faulty

Use heuristics based on element relationships

- Weighted prioritization score

\[ r(e) = w \times C(e) + x \times O(e) + y \times D(e) + z \times P(e) \]

- Lower the score, higher the likelihood of being faulty
3.1 Contained Elements ($C$)

\[
C(e) = \begin{cases} 
1 & \text{if } e.\text{parent} \in \text{resultSet} \land e.\text{allSiblings} \subset \text{resultSet} \\
0 & \text{otherwise}
\end{cases}
\]

Expected appearance

Actual appearance
3.2 Overlapped Elements ($\mathcal{O}$)

\[
\mathcal{O}(e) = \begin{cases} 
1 & \text{if } 1 \leq |e\text{.children} \cap \text{resultSet}| < |e\text{.children}| \\
0 & \text{otherwise}
\end{cases}
\]

**Expected appearance**

- **parent**
  - child1
  - child2

**Actual appearance**

- **parent**
  - child1
  - child2
3.3 Cascading ($\mathcal{D}$)

$$
\mathcal{D}(e) = \begin{cases} 
1 & \text{if croppedImage}(e) \text{ is a sub-image of Oracle} \\
0 & \text{otherwise}
\end{cases}
$$

Expected appearance

```
+----+----+----+
|    | element |    |
+----+----+----+
| 1  | element |
+----+----+----+
| 2  | element |
+----+----+----+
| 3  | element |
+----+----+----+
```

Actual appearance

```
+----+----+----+
|    | element |    |
+----+----+----+
| 1  | element |
+----+----+----+
| 2  | element |
+----+----+----+
| 3  | element |
+----+----+----+
```

Expected appearance

Actual appearance
3.4 Pixels Ratio ($P$)

$$P(e) = 1 - \frac{|\{d \mid d \in \text{diffPixels} \land d \text{ within } e.\text{area}\}|}{e.\text{width} \times e.\text{height}}$$

- Child pixels ratio = 100%
- Parent pixels ratio = 20%
P3. Result Set Processing - Example

Report

Cluster A
/html/body/table/.../img
/html/body
/html/body/table
:
5. /html/body/table
6. /html/body
/html/body/table/.../img

Cluster B
Cluster C
Cluster D
Cluster E
Empirical Evaluation

• RQ1: What is the accuracy of our approach for detecting and localizing presentation failures?

• RQ2: What is the quality of the localization results?

• RQ3: How long does it take to detect and localize presentation failures with our approach?
Experimental Protocol

• Approach implemented in “WebSee”
• Five real-world subject applications
• For each subject application
  – Download page and take screenshot, use as the oracle
  – Seed a unique presentation failure to create a variant
  – Run WebSee on oracle and variant
# Subject Applications

<table>
<thead>
<tr>
<th>Subject Application</th>
<th>Size (Total HTML Elements)</th>
<th>Generated # test cases</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gmail</td>
<td>72</td>
<td>52</td>
</tr>
<tr>
<td>USC CS Research</td>
<td>322</td>
<td>59</td>
</tr>
<tr>
<td>Craigslist</td>
<td>1,100</td>
<td>53</td>
</tr>
<tr>
<td>Virgin America</td>
<td>998</td>
<td>39</td>
</tr>
<tr>
<td>Java Tutorial</td>
<td>159</td>
<td>50</td>
</tr>
</tbody>
</table>
RQ1: What is the accuracy?

- **Detection accuracy**: Sanity check for PID
- **Localization accuracy**: % of test cases in which the expected faulty element was reported in the result set

<table>
<thead>
<tr>
<th>Platform</th>
<th>Localization Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Java Tutorial</td>
<td>94%</td>
</tr>
<tr>
<td>Virgin America</td>
<td>97%</td>
</tr>
<tr>
<td>Craigslist</td>
<td>90%</td>
</tr>
<tr>
<td>USC CS Research</td>
<td>92%</td>
</tr>
<tr>
<td>Gmail</td>
<td>92%</td>
</tr>
</tbody>
</table>

Localization accuracy = 93%
RQ2: What is the quality of localization?

<table>
<thead>
<tr>
<th>Source</th>
<th>Result Set Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Java Tutorial</td>
<td>8 (5%)</td>
</tr>
<tr>
<td>Virgin America</td>
<td>49 (5%)</td>
</tr>
<tr>
<td>Craigslist</td>
<td>32 (3%)</td>
</tr>
<tr>
<td>USC CS…</td>
<td>17 (5%)</td>
</tr>
<tr>
<td>Gmail</td>
<td>12 (16%)</td>
</tr>
</tbody>
</table>

Result Set Size

Distance = 6

Rank = 4.8 (2%)

faulty element not present
RQ3: What is the running time?

7 sec

87 sec

3 min

Sub-image search for cascading heuristic

**P1: Detection**

**P2: Localization**

**P3: Result Set Processing**

54% 21% 25%
Comparison with User Study

• Graduate-level students
• Manual detection and localization using Firebug

• Time
  – **Students**: 7 min
  – **WebSee**: 87 sec
Case Study with Real Mockups

- Three subject applications
- 45% of the faulty elements reported in top five
- 70% reported in top 10
- Analysis time similar
Summary

- Technique for automatically detecting and localizing presentation failures
- Use computer vision techniques for detection
- Use rendering maps for localization
- Empirical evaluation shows positive results
Thank you

Detection and Localization of HTML Presentation Failures Using Computer Vision-Based Techniques

Sonal Mahajan and William G. J. Halfond

spmahaja@usc.edu
halfond@usc.edu
Normalization Process

• Pre-processing step before detection

1. Browser window size is adjusted based on the oracle
2. Zoom level is adjusted
3. Scrolling is taken care of
Difference with XBT

- XBT use DOM comparison
  - Find matched nodes, compare them
- Regression debugging
  - Correct bug, refactor HTML (e.g. `<table>` to `<div>` layout)
  - DOM significantly changed
    - XBT cannot find matching DOM nodes, not accurate comparison
- Mockup Driven Development
  - No “golden” version of page (DOM) exists
  - XBT techniques cannot be used
- Our approach
  - Uses computer vision techniques for detection
  - Applies to both scenarios
Pixel-to-pixel Comparison

Oracle

Test Webpage Screenshot
Pixel-to-pixel Comparison

98% of the entire image is shown in difference!

Difference image

- Difference pixel
- Matched pixel
P1. Detection

• Find visual differences (presentation failures)

• **Simple approach:** strict pixel-to-pixel equivalence comparison

---

**Analyze using computer vision techniques**

• **Our approach:** Perceptual image differencing (PID)
Perceptual Image Differencing

Difference image

- Difference pixel
- Matched pixel
High fidelity mockups… reasonable?

Key Responsibilities / Performance Requirements:

- Generate pixel-perfect web pages from high fidelity mock ups created in Photoshop and Fireworks.
High fidelity mockups... reasonable?

- Expert at coding semantic HTML markup
- Ability to develop pixel perfect sites that meet the design requirements
- Experience building responsive websites
High fidelity mockups… reasonable?

- Gathering, interpreting requirements from clients and designers and translating them into CMS based websites with pixel-perfect fidelity to Photoshop comps.
High fidelity mockups… reasonable?

- Proficient in converting PSD compositions into HTML and CSS markup
- Must have high attention to detail, capable of pixel-perfect page construction that performs well across browsers.
High fidelity mockups... reasonable?

Front-end Developer
for Guidance Solutions, Inc in Marina del Rey, CA

- Interpret existing site design and style guides by creating pixel perfect web pages from Photoshop files
High fidelity mockups… reasonable?

- Code pixel-perfect, cross-browser compatible templates using HTML 5 and CSS, from designs provided by our internal design team or external agencies.
High fidelity mockups… reasonable?

- Be able to create pixel perfect sites/pages from approved creative designs, and also be able to offer feedback to the team when UI/UX elements need to be altered in order to adhere to development best practices
High fidelity mockups… reasonable?

Responsible for creating pixel perfect templates based on designer mock-ups, using HTML, CSS, and JavaScript.
The Front-end Web Development Team for Sony Electronics in San Diego, CA, is currently looking for creative and talented Web Developer’s that will be responsible for translating visual designs into pixel perfect, standards-compliant web pages for the Sony Store Online e-commerce website.
High fidelity mockups… reasonable?

Ability to develop pixel perfect implementations of Photoshop comps
High fidelity mockups… reasonable?

Our new web developer will work closely with our Ecommerce UX, Design, and Development Teams to transform mock-ups into cross-platform pixel-perfect web interfaces--achieving well executed, high-performing, scalable, and easily maintained results.
High fidelity mockups... reasonable?

Careers at Toaster: Front End Developer

- Excellent eye for layout and ability to translate PSD files into pixel perfect HTML/CSS code