Pinpoint Analysis of Software Usability

Dan Tamir
Department of Computer Science
Texas State University-San Marcos
Dan Tamir, Associate Professor, Computer Science, Texas State University

• **Education:**
  – BS & MS-EE (BGU), PhD-CS (FSU)

• **Professional experience:**
  – Florida Tech, Motorola/Freescale, TX State

• **Areas of Interest:**
  – Incremental classification of Big Data
  – Disaster & Pandemic preparedness & mitigation via anomaly detection,
  – image processing,
  – usability

• **Recent funding:**
  – Automating bridge inspection-feasibility study (TxDOT)
  – Power aware Task Scheduling (Semi-conductor Research Consortium)
  – Pinpointing of Software Usability Issues (Emerson – Process Control)
  – Laser lithography on non-flat surface (NSF)
  – Introducing parallel processing early in the curriculum (NSF)
Agenda

• Effort Base Usability Evaluation,
• Pinpoint analysis,
• Pattern Recognition tools used
• Experiments
  • Setup
  • Procedures
  • evaluation methodology
• Experiments, results, results’ analysis
• Example – Non Destructive UI
Measuring Usability
Usability

• The ease with which a user can learn to operate, prepare-inputs for, and interpret outputs of a system or component.” (IEEE 1990)

• “The Extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency, and satisfaction in a specified context of use.” (ISO 9241-1, 1998)

• “The capability of the software product to be understood, learned, used, and attractive to the user, when used under specified conditions” (ISO 9126-1, 2001).
Usability Attributes

- Effectiveness – The product enables users to achieve specified goals with accuracy and completeness in a specified context.
- Efficiency – The resources expended in relation to the accuracy and completeness with which the user achieves goals.
- Satisfaction – The comfort and acceptability of use.
- Productivity – The product enables users to expend appropriate amount of resources in relation to the effectiveness.
Attributes (continued)

• Understandability - The ability of a user to understand the capabilities of the software.
• Learnability - The ease with which a user learns to use the software.
• Operability - The capability of a user to use the software to accomplish a specific goal.
• Attractiveness - The appeal of the software to a user.
Observations

• Usability is inversely proportional to effort
  – User effort is related to manual effort – e.g., number of mouse clicks, number of key-board clicks, mouse path traversed.

• A set of identical independent ("iid") experiments on a single scenario can be used to measure learnability and operability

• Eye tracking can be used to provide additional measures of physical and manual effort
Traditional Learning Curve

\[
\text{Time-on-Task} = \alpha X^\beta + \varepsilon
\]
Effort-based Usability Model

*Based on ISO/IEC 9126-1:2001 Standard*
Eye Tracker Hardware

Eye tracker

Chin rest
Fixations and Saccades

- When performing a task, fixations and saccades can reflect effort expended.
  - Greater effort =
    - Longer fixation duration
    - More fixations
    - Longer saccade length
    - More saccades
Travel Reservation Experiment: Time on Task

\[ y = 622.65x^{-0.217} \]
\[ R^2 = 0.7221 \]

\[ y = 309.1x^{-0.278} \]
\[ R^2 = 0.8792 \]
Usability Requirements Specifications
Examples of usability requirements of VacationPro

• Effectiveness – At least 90% of the users will complete at least 90% of the task of hotel reservation under a specific set of required amenities with 90% accuracy of compliance with the requirements, in less than 10 minutes.

• Efficiency – Given $x$ productive-users attempting $y$ tasks of hotel reservation under a specific set of required amenities, at least 90% of the users will expand no more than 120% of the resources expended by experts attempting these $y$ tasks under the specified set of constraints.
Usability Requirements of VacationPro

- **Satisfaction** – The mean score on the SUMI scale will be greater than 50.

- **Productivity** - Given $x$ productive-users attempting $y$ tasks of flight reservation under a specific set of budget constraints at least 90% of the users will expand no more than 120% of the resources expended by experts attempting these $y$ tasks under the specified set of constraints (quite similar to efficiency).

- **Understandability** – Productive users will have less than 5% of errors of type 1 (assuming functionality that is not available in the system) and less than 5% errors of type 2 (insufficient knowledge of available system functionality).
Usability Requirements

• Learnability - The average novice user will reach the level of productive user after $x$ number of executions of each specific scenario based independent identical set of tasks.

• Operability - (quite similar to efficiency).

• Attractiveness - At least 95% of the users that have any experience with the system will rank the system appeal level at 8 or above on a scale of 1 (low attractiveness) to 10 (high attractiveness).
Usability Testing
Examples of Requirements-Based Testing Procedures (VacationPro)

- Effectiveness, Efficiency, and Operability - Measure the average ToT of $x$ productive users attempting $y$ independent identical tasks of hotel reservation under a specific set of amenities constraints.

- Satisfaction – Administer the SUMI tests. Alternatively, assess user satisfaction via one way mirrors.
Examples of Requirements-Based Testing Procedures

- Productivity - Measure the average ToT of $x$ productive users attempting $y$ tasks of hotel reservation under a specific set of amenities constraints and compare it to the ToT of an expert.

- Understandability – Administrate a set of tests to check the average rate of errors of type 1 and type 2 in associating functionality to the system by a set of $x$ productive users.
Requirements-Based Testing

Procedures

• Learnability – Plot the average learning (effort) curve (e.g., using eye path traversed as the effort measure) of $x$ novice users. Identify the point of reaching a productive level state for each user.

• Attractiveness – Using questionnaires assess the ranking of appeal of the system by a set of users with any level experience with the system.
Pilot Project
Emerson / TxState Usability Experiment

• Purpose
  – Pilot Study to determine the usefulness of the Texas State University methodology in measuring aspects of Usability in Emerson products

• Primary Goal
  – Compare the usability of a limited set of tasks in two versions of Control Studio referred to as System A and System B
Scenario-based Test Design

- The test consisted of 15 repetitive tasks.
- Each task followed the same general workflow,
  - However, function blocks, parameters, and properties being worked on, were varied.

- The task instructions were written in general terms such as “Add an AI block”, but did not specify how to carry out the work.
Scenario-based tasks used in the Experiments

Appendix C Tasks

**TASK 1**

*Start*

1. Delete block PT3-15 from the Distillation Column COLUMN1.
2. Add an Analog Output to the right of the block PIC3-15 and name it as VENT_VALVE.
3. Make the following connections -
   a. VENT_VALVE OUT to PIC3-15 BKCAL_IN and set the connection as feedback
   b. PIC3-15 OUT to VENT_VALVE CAS_IN
4. Transfer the changes to the Controller Simulator. Change Control Studio to view the information from the Controller Simulator.
5. Change the PIC3-15 Pressure control set point (SP) to 25.
6. Change Control Studio to view the information in the Configuration Database.
7. Upload and save the changes

*End*

**TASK 2**

*Start*

1. Delete block LIC3-16_RSP from the Distillation Column COLUMN1.
2. Add an Analog Input to the left of the block PIC3-15 and name it as PT3-15.
3. Make the following connections -
   a. PT3-15 OUT to PIC3-15 IN and set the connection as feedback
4. Transfer the changes to the Controller Simulator. Change Control Studio to view the information from the Controller Simulator.
5. Change the VENT_VALVE SP_HL_LIM to 85
6. Change Control Studio to view the information in the Configuration Database.
7. Upload and save the changes

*End*
Mean TOT: System A

Mean Time-On-Task

Minutes

Task Number

System A
Standard Deviation for ToT in System A

Mean Time-On-Task

Minutes

Task Number

System A
Power Curve Matching to ToT of System A

Mean Time-On-Task

\[ y = 3.8080x^{-0.2690} \]

\[ R^2 = 0.8219 \]
Mean TOT: System B

Mean Time-On-Task

<table>
<thead>
<tr>
<th>Task Number</th>
<th>Minutes</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
</tr>
<tr>
<td>7</td>
<td>2</td>
</tr>
<tr>
<td>8</td>
<td>2</td>
</tr>
<tr>
<td>9</td>
<td>2</td>
</tr>
<tr>
<td>10</td>
<td>2</td>
</tr>
<tr>
<td>11</td>
<td>2</td>
</tr>
<tr>
<td>12</td>
<td>2</td>
</tr>
<tr>
<td>13</td>
<td>2</td>
</tr>
<tr>
<td>14</td>
<td>2</td>
</tr>
<tr>
<td>15</td>
<td>2</td>
</tr>
</tbody>
</table>
Standard Deviation for ToT in System B

Mean Time-On-Task

Task Number

Minutes

System B
Power Curve Matching to ToT of System B

Mean Time-On-Task

System B:

\[ y = 3.9822x^{-0.2777} \]

\[ R^2 = 0.8609 \]
Overall Learnability

Mean Time-On-Task

System A:
y = 3.8294x^{-0.2320}
R^2 = 0.8008

System B:
y = 3.9822x^{-0.2777}
R^2 = 0.8609
Physical Effort

Mean Eye Path Traversed

System A:
\[ y = 1,446.7236x^{0.2387} \]
\[ R^2 = 0.7327 \]

System B:
\[ y = 1,962.7388x^{0.3365} \]
\[ R^2 = 0.8332 \]
Physical Effort

Mean Mouse-Path Traversed

System A:
\[ y = 25,694.2029x^{-0.1227} \]
\[ R^2 = 0.5831 \]

System B:
\[ y = 34,390.1019x^{-0.2426} \]
\[ R^2 = 0.6913 \]
Experiment Conclusions

- A methodology involving eye tracking is a viable tool for objectively measuring usability.

- After Learning point is reached, both System A and B have very similar usability characteristics.

- People are able to learn to use the application with the updated user interface.

- [After moderate training] student performance is close to “real user’s” performance.
Current / Next Phases

• Phase 2
  – Analysis of additional scenarios using current Emerson software and prototypes of “next generation software”.

• Phase 3
  – Pinpoint analysis
Pinpoint Analysis
Pinpoint Analysis

\[ R = \begin{bmatrix} r_1 \\ r_2 \\ \cdot \\ \cdot \\ r_j \end{bmatrix} \]

\[ R_1 = \text{Average Saccade Amplitude} \]

\[ R_2 = \text{Average Fixation duration} \]

\[ R_j = \text{Total Time on Task} \]
Pinpoint Analysis

- Segment the data
- Use pattern recognition techniques to identify excessive-effort segments
  - Thresholding
  - Clustering (K-means)
    - Exhaustive feature selection
    - Principle component analysis
- Video clips corresponding to identified excessive-effort segments are further analyzed to spot usability issues
Pinpoint Analysis

Definitions

• **Pinpoint Analysis**: Identifying and pinpointing issues with the interface.

• **Inter-pinpoint Analysis**: Identifying issues with tasks in a specific system.

• **Intra-pinpoint Analysis**: Identifying issues within tasks in a specific system.
Pinpoint Analysis Example

Example: Through inter-pinpoint analysis we can identify tasks that present usability issues (outliers) and select those tasks for intra-pinpoint analysis attempting to understand the root cause of the issues.

\[
y = 7.4958x^{-0.321} \\
R^2 = 0.962
\]
Pattern Recognition

• Assignment of *labels* to a given input value, or *instance*, according to a specific algorithm.

• Can be categorized based on the learning procedure. Two type:

  *Supervised learning*
  - Training data properly labeled by hand with the correct output, has been provided.
  - Learning procedure generates a model for classification

  *Unsupervised learning*
  - Training data that has not been hand-labeled
  - Attempts to find inherent patterns to determine the correct classification value for new data instances

• Algorithms differ in the way inference is performed like – based on probability, fuzzy logic, on non parametric clustering.
Pattern Recognition Operations

The following are various relevant pattern recognition operations:

- Segmentation
- Feature Extraction and Feature Selection
- Principal Component Analysis (PCA)
- Clustering
- Applying a Threshold
1) Segmentation
   • Definition of patterns
   • Segments of user activity records serve as patterns
   • A segment is the time between two consecutive keyboard/mouse clicks

2) Feature Extraction and Feature Selection
   • Patterns subject to classification are represented as set of measurements referred to as features
   • Selecting a subset of relevant features is called as Feature selection.
   • Feature selection algorithms attempt to reduce the dimensionality of the feature space and reduce the complexity
Pattern Recognition Operations (Continued)

• Feature Extraction and Feature Selection

_Exhaustive Search:_
- Brute-force feature selection method
- All possible subsets of the features are exhaustively evaluated and the best subset is selected.
- The number of combinations of $R$ objects from a set of $N$ features is $\frac{N!}{R!(N-1)!}$

_Heuristic/Suboptimal Search:_
- Selection by making an educated guess and finding out if the selection yields good results.
- A good alternative where an exhaustive search is impractical.
Pattern Recognition Operations (Continued)

3) Principal Component Analysis (PCA)
   • Unsupervised learning procedure
   • Coordinate transformation that de-correlates the data and orders the information (or variance) associated with the data in the axes of the new space in a monotonically decreasing fashion.
   • Information associated with the data is concentrated in the first few components of the new space.
   • Each principal component is a linear combination of the original variables.

4) Applying a Threshold
   • Classify input data based on a threshold value, like average.
   • Input Values > threshold are put into one group while input values < threshold are classified into a second group.
   • Limited to one dimensional data.
5) Clustering

- Unsupervised learning procedure.
- Assignment of a set of patterns into subsets (called clusters) such that patterns in the same cluster are similar in some sense.
- **K-means algorithm**: Partition \( n \) observations into \( k \) clusters in which each observation belongs to the cluster with the nearest mean
- Goal: Attempts to minimize the mean square distance between patterns and cluster centers.
- Algorithm:
Comparison of Pattern Recognition Operations

Difference between PCA and Feature Selection:

- Following PCA, the resulting features are different than the original features. They do not correspond directly to original set of measurements.
- features left after feature selection are simply a subset of the original features

Difference between Thresholding and Clustering:

- A threshold is applied only on individual features or linear combination of features.
- Clustering is applied on multi-dimensional data.
The experiment procedure includes the following three main phases:

- **Phase 1 - Data Gathering**
- **Phase 2 – Reduction** (includes segmentation, data reduction, feature extraction and selection)
- **Phase 3 – Identification of Excessive Segments**
Experiment Test Procedure (Continued)

Phase 1 - Data Gathering
- Tasks designed
- Experiments conducted
- Data collected throughout the interaction process: eye data, keyboard, mouse activities are logged by an eye tracker.

Phase 2 – Data Processing
- Data reduction
- Segmentation
- Feature extraction
Phase 3 – Identification of Excessive Effort Segments

Applying threshold:
- threshold value is calculated
- feature value < threshold value → classified as non-excessive
- feature value > threshold value → classified as excessive

Applying K-means:
- the segments are grouped into clusters.
- cluster centers used to identify excessive effort cluster.
- All segments that fall in the excessive cluster are segments exhibit excessive effort behavior and vice versa.
Phase 3 – Identification of Excessive Effort Segments (continued)

PCA:

- The first, second, and third principal components are obtained for the feature data.
- A threshold classification is applied on the first principal component.
- K-means clustering is applied on the first, second, and third components to classify the segments into excessive or non-excessive.

- Identification of segments is automated by a program referred to as “Software Program” and the classification is called automatic classification.
- At the end of phase 3 excessive effort segments are identified.
Manual Classification

- **Idle behavior segments**: idle behavior is due to system response
- **Excessive effort segments**: segments without any useful user action are classified as excessive effort segments.
- **Non-Excessive effort segments**: segments with useful action that result in task completion are classified as non-excessive segments.
- **Off screen behavior segments**: Intervals of time where the subject’s view is not within the screen dimensions for more than one second, with no meaningful user action are classified as off screen behavior segments.
- **Attention segments**: segments with frequent off screen behavior, frequent mouse/keyboard clicks are classified as attention segments.
Results’ Verification

- The number of E vs. E, E vs. NE, NE vs. E, NE vs. NE are calculated
- Graphs are plotted.

Sample Result File

<table>
<thead>
<tr>
<th>Number of Fixations</th>
<th>Segment Start Time</th>
<th>Segment End Time</th>
<th>Manual Classification</th>
<th>Tool Classification</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td>551</td>
<td>NE</td>
<td>NE</td>
</tr>
<tr>
<td></td>
<td>551</td>
<td>1451</td>
<td>NE</td>
<td>E</td>
</tr>
<tr>
<td></td>
<td>1451</td>
<td>3088</td>
<td>NE</td>
<td>E</td>
</tr>
<tr>
<td></td>
<td>3088</td>
<td>5640</td>
<td>NE</td>
<td>E</td>
</tr>
<tr>
<td></td>
<td>5640</td>
<td>5640</td>
<td>NE</td>
<td>NE</td>
</tr>
<tr>
<td></td>
<td>5640</td>
<td>10880</td>
<td>E</td>
<td>E</td>
</tr>
<tr>
<td></td>
<td>10880</td>
<td>11296</td>
<td>NE</td>
<td>NE</td>
</tr>
<tr>
<td></td>
<td>11296</td>
<td>11488</td>
<td>NE</td>
<td>NE</td>
</tr>
<tr>
<td></td>
<td>11488</td>
<td>11681</td>
<td>NE</td>
<td>NE</td>
</tr>
<tr>
<td></td>
<td>11681</td>
<td>11921</td>
<td>NE</td>
<td>NE</td>
</tr>
<tr>
<td></td>
<td>11921</td>
<td>17840</td>
<td>NE</td>
<td>NE</td>
</tr>
<tr>
<td></td>
<td>17840</td>
<td>17840</td>
<td>NE</td>
<td>NE</td>
</tr>
<tr>
<td></td>
<td>17840</td>
<td>20921</td>
<td>NE</td>
<td>E</td>
</tr>
<tr>
<td></td>
<td>20921</td>
<td>22670</td>
<td>A</td>
<td>E</td>
</tr>
<tr>
<td></td>
<td>22670</td>
<td>22670</td>
<td>A</td>
<td>NE</td>
</tr>
<tr>
<td></td>
<td>22670</td>
<td>28409</td>
<td>A</td>
<td>E</td>
</tr>
<tr>
<td></td>
<td>28409</td>
<td>30090</td>
<td>A</td>
<td>E</td>
</tr>
<tr>
<td></td>
<td>30090</td>
<td>31731</td>
<td>A</td>
<td>E</td>
</tr>
<tr>
<td></td>
<td>31731</td>
<td>33722</td>
<td>A</td>
<td>E</td>
</tr>
<tr>
<td></td>
<td>33722</td>
<td>37232</td>
<td>A</td>
<td>E</td>
</tr>
<tr>
<td></td>
<td>37232</td>
<td>37584</td>
<td>A</td>
<td>NE</td>
</tr>
<tr>
<td></td>
<td>37584</td>
<td>37728</td>
<td>A</td>
<td>NE</td>
</tr>
<tr>
<td></td>
<td>37728</td>
<td>37904</td>
<td>A</td>
<td>NE</td>
</tr>
<tr>
<td></td>
<td>37904</td>
<td>38416</td>
<td>A</td>
<td>NE</td>
</tr>
<tr>
<td></td>
<td>38416</td>
<td>40892</td>
<td>NE</td>
<td>NE</td>
</tr>
</tbody>
</table>
Type-I Errors

- segments that show non-excessive effort per manual classification but identified as excessive effort segments by the software program regarded as false positive or type-I error segments.
  - The software program is highlighting some extra segments for further review

Type-II Errors

- Segments that show excessive effort per manual classification but identified as non-excessive effort segments by the software program are regarded as false negative or type-II error segments.
  - The software program missed segments that require manual inspection.
Inspection Time

Inspection time

- The total time of segments classified as excessive by the software program
- The sum of the time interval of each excessive effort segment.

In this paper, type-II errors and inspection time are considered as the most important factors for analyzing experiment results.
Experiments
Experiments

Experiment 1: Identifying excessive effort segments using thresholding

Experiment 2: Identifying excessive effort segment using K-means clustering

Experiment 3: Identifying excessive effort segments using thresholding applied to the first principal components

Experiment 4: Identifying excessive effort segments using K-means clustering on first, second, and third principal components.
Experiment 1

Identifying excessive effort segments using the threshold method

Feature set
- Number of fixations
- Average fixation duration
- Number of saccades
- Average saccade amplitude
- Eye path traversed.
Observation:
- ‘number of fixations’ performs well in terms of type-II errors.
- ‘Average fixation duration’ has very small inspection time. But, 15.05% for type-II errors.
- ‘Average saccade amplitude’ has minimum inspection time and acceptable type-II errors.
Data File 5

Observation:
- ‘number of fixations’ performs well in terms of type-II errors.

Observation:
- ‘Average fixation duration’ has very small inspection time and high type-II errors
- ‘Number of fixations’ has low inspection time and type-II errors within an acceptable range.
Experiment 2


Feature set

- Subset 1: Number of fixations
- Subset 2: Number of saccades
- Subset 3: Eye path traversed
- Subset 4: Number of fixations, number of saccades, eye path traversed
- Subset 5: Number of fixations, number of saccades, eye path traversed, average fixation
Observation: feature subset 1 performs well in terms of type-II errors.

Observation: Subset 3, with ‘eye path traversed’ as a feature value has very small inspection time and type-II errors in an acceptable range.
Observation:

- Feature subset 1 and subset 5 have minimum type-II errors.
- Feature subsets 3 and 4 show a relatively low value of inspection time.
- Percentage of type-II errors is 7.69% for subset 3 and 4.76% for feature subset 4.
- The feature value with lower type-II errors and lower percentage of time of segments classified as excessive is feature subset 3.
Experiment 3

Identifying excessive effort segments using principal component analysis and thresholding

Feature set:
1st principal component
Data File Analysis

Observation:
- relatively low values for type-II errors for all data files.

Observation:
- Inspection time is relatively high when applying thresholding on first principal component.
Experiment 4

Identifying excessive effort segments using K-means clustering on principal components

Feature set:
1st principal component
1st principal component, 2nd principal component
1st principal component, 2nd principal component, 3rd principal component
Observation

- All features have the same type-I and type-II errors

Observation

- Relatively low inspection time
- Type-II errors are not within acceptable limit.
Observation

- All features have same type-I and type-II errors

- Very low inspection time
- Type-II errors are 5.3% within acceptable limit.
Result Analysis

Criteria for success

1) The number of *type-II errors*

2) A minimal *time to investigate* usability issues with a level of 15% of type-II errors. This level is considered acceptable.
# Experiment 1 Analysis

<table>
<thead>
<tr>
<th>Feature value</th>
<th>avg. # of excessive effort segments</th>
<th>avg. total no of segments</th>
<th>avg. % type-I errors</th>
<th>avg. % type-II errors</th>
<th>avg. % of total errors</th>
<th>avg. Inspection time</th>
<th>avg. Inspection time as a % of total time</th>
</tr>
</thead>
<tbody>
<tr>
<td># Fix</td>
<td>17.2</td>
<td>95</td>
<td>28.4</td>
<td>3.3</td>
<td>31.7</td>
<td>2.7</td>
<td>62.1</td>
</tr>
<tr>
<td>Avg Fix Dur</td>
<td>18.2</td>
<td>95</td>
<td>29.5</td>
<td>9.9</td>
<td>39.4</td>
<td>1.6</td>
<td>37.4</td>
</tr>
<tr>
<td>#Sacc</td>
<td>32</td>
<td>95</td>
<td>21.8</td>
<td>10.5</td>
<td>32.2</td>
<td>2.9</td>
<td>64.1</td>
</tr>
<tr>
<td>Sacc Amp</td>
<td>17.6</td>
<td>95</td>
<td>29.1</td>
<td>4.6</td>
<td>33.7</td>
<td>2.5</td>
<td>56.4</td>
</tr>
<tr>
<td>Eye Path</td>
<td>17.8</td>
<td>95</td>
<td>25.7</td>
<td>5.1</td>
<td>30.8</td>
<td>2.6</td>
<td>57.7</td>
</tr>
</tbody>
</table>

- The metric ‘number of fixations,’ gives good results in terms of type-II errors but, the average inspection time is relatively high.
- The metric ‘average fixation duration’ performs well in terms of minimal inspection time with an acceptable value of 9.8% for type-II errors.
Experiment 1 Analysis (Continued)

- The metric ‘eye path traversed’ has minimum total errors.
- The inspection time is not completely correlated to type-I errors.
- Segments classified as excessive are different for each feature value.
- The percentages of total errors for each feature value are in close proximity to each other, but inspection times vary.
## Experiment 2 Analysis

<table>
<thead>
<tr>
<th>Feature value</th>
<th>avg. # of excessive effort segments</th>
<th>avg. total no of segments</th>
<th>avg. % type-I errors</th>
<th>avg. % type-II errors</th>
<th>avg. % of total errors</th>
<th>avg. Inspection time</th>
<th>avg. Inspection time as a % of total time</th>
</tr>
</thead>
<tbody>
<tr>
<td>#fix</td>
<td>29.1</td>
<td>95</td>
<td>27.2</td>
<td>6.6</td>
<td>33.9</td>
<td>2.4</td>
<td>56.2</td>
</tr>
<tr>
<td>#sacc</td>
<td>23.5</td>
<td>95</td>
<td>17.8</td>
<td>8.9</td>
<td>26.7</td>
<td>2.0</td>
<td>45.1</td>
</tr>
<tr>
<td>eye path</td>
<td>19.7</td>
<td>95</td>
<td>18.0</td>
<td>10.1</td>
<td>28.1</td>
<td>1.6</td>
<td>37.5</td>
</tr>
<tr>
<td>#fix, #sacc, eye path</td>
<td>23.2</td>
<td>95</td>
<td>18.3</td>
<td>8.6</td>
<td>26.9</td>
<td>1.9</td>
<td>44.5</td>
</tr>
<tr>
<td>#fix, #sacc, eye path, avg. fix dur., avg. sacc amp.</td>
<td>29.2</td>
<td>95</td>
<td>32.6</td>
<td>5.4</td>
<td>38.0</td>
<td>2.5</td>
<td>56.3</td>
</tr>
</tbody>
</table>

➢ The feature subset- ‘number of fixations,’ ‘number of saccades,’ ‘eye path traversed,’ ‘average fixation duration,’ and ‘average saccade amplitude,’ gives good results in terms of type-II.
Experiment 2  Analysis (Continued)

➢ The metric ‘eye path traversed’ performs well in terms of minimal inspection time with an acceptable value of 10.1% for type-II errors.

➢ The metric ‘number of fixations’ has minimum total number of errors but a relatively high inspection time.

➢ The number of excessive effort segments for ‘number of fixations’ and the feature subset with the following features – ‘number of saccades,’ ‘eye path traversed,’ ‘average fixation duration,’ and ‘average saccade amplitude’ are the same. However, the inspection times vary
Experiment 3 Analysis

<table>
<thead>
<tr>
<th>Feature value</th>
<th>avg. # of excessive effort segments</th>
<th>avg. total no of segments</th>
<th>avg. % type-I errors</th>
<th>avg. % type-II errors</th>
<th>avg. % of total errors</th>
<th>avg. Inspection time</th>
<th>avg. Inspection time as a % of total time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1st principal components</td>
<td>16.6</td>
<td>95</td>
<td>27.5</td>
<td>4.1</td>
<td>31.6</td>
<td>2.7</td>
<td>61.2</td>
</tr>
</tbody>
</table>

- Results are comparable to the results obtained from Experiment 1
- Type-II errors for number of fixations is 3.3%
Experiment 3 Analysis (Continued)

- The number of type-II errors for the first principal component is 4.1%
- The inspection time for the first principal component and for the average fixation duration are 2.7 and 1.6 minutes respectively.
- A threshold on the metric ‘average fixation duration’ performs better than first principal component in terms of lower inspection time and an acceptable 9.8% for type-II errors.
# Experiment 4 Analysis

<table>
<thead>
<tr>
<th>Feature value</th>
<th>avg. # of excessive effort segments</th>
<th>avg. total no of segments</th>
<th>avg. % type-I errors</th>
<th>avg. % type-II errors</th>
<th>avg. % of total errors</th>
<th>avg. Inspection time</th>
<th>avg. Inspection time as a % of total time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1st, 2nd &amp; 3rd principal components</td>
<td>28.6</td>
<td>95</td>
<td>24.4</td>
<td>12.6</td>
<td>37.0</td>
<td>2.0</td>
<td>43.6</td>
</tr>
</tbody>
</table>

- The average value of type-II error is relatively high.
- The average inspection time is only 1.96%.
Summary

Type-II errors:
- Applying a threshold on the ‘number of fixations’ yields the best results in terms of type-II errors, followed by a threshold on the first principal component.
- Applying K-means clustering on feature subset with features: ‘number of fixations,’ ‘number of saccades,’ ‘average saccade amplitude,’ ‘average fixation duration,’ and ‘eye path traversed’ ranks third.

Inspection Time:
- Applying K-means clustering on number of saccades yields good results.
- Followed by thresholding on ‘average fixation duration’
Conclusion

• The proposed framework enables software developers to efficiently identify usability issues thereby optimizing time spent on usability testing.

• Excessive effort segments, which typically relate to usability issues, are identified by applying pattern recognition techniques.

• Usability testing can be reduced by 40%.
Recommendations for Future Research

• Equal time slicing of user’s software interaction session can be used as a segmentation method.

• Further refinement of the pattern recognition techniques to improve the errors and inspection time can be considered.

• Another direction for future research can be to automate some of the manual steps in this process.
The Sentic Mouse

Physiological Emotion applications

- MIT Affective Computing Lab’s Affective Tangibles Program
- Mouse behaviors – number of mouse clicks, duration of mouse clicks
Non Distractive User Interface
Non Distractive User Interface
Non Distractive User Interface
Research Implementation Issues

• Voice Input / Output
• Intelligent Crawling
  ▪ Data Mining
    ▪ Incremental Clustering
  ▪ Prediction

▪ Usage by Driver
▪ Can it be used by Visually challenged people
Snapshots

Figure 4: Initial screenshot of developed interface
Snapshots

The initial page is:

This page shows three URLs at the top left.