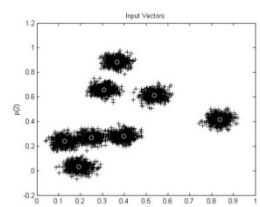
Pinpoint Analysis of Software Usability

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.

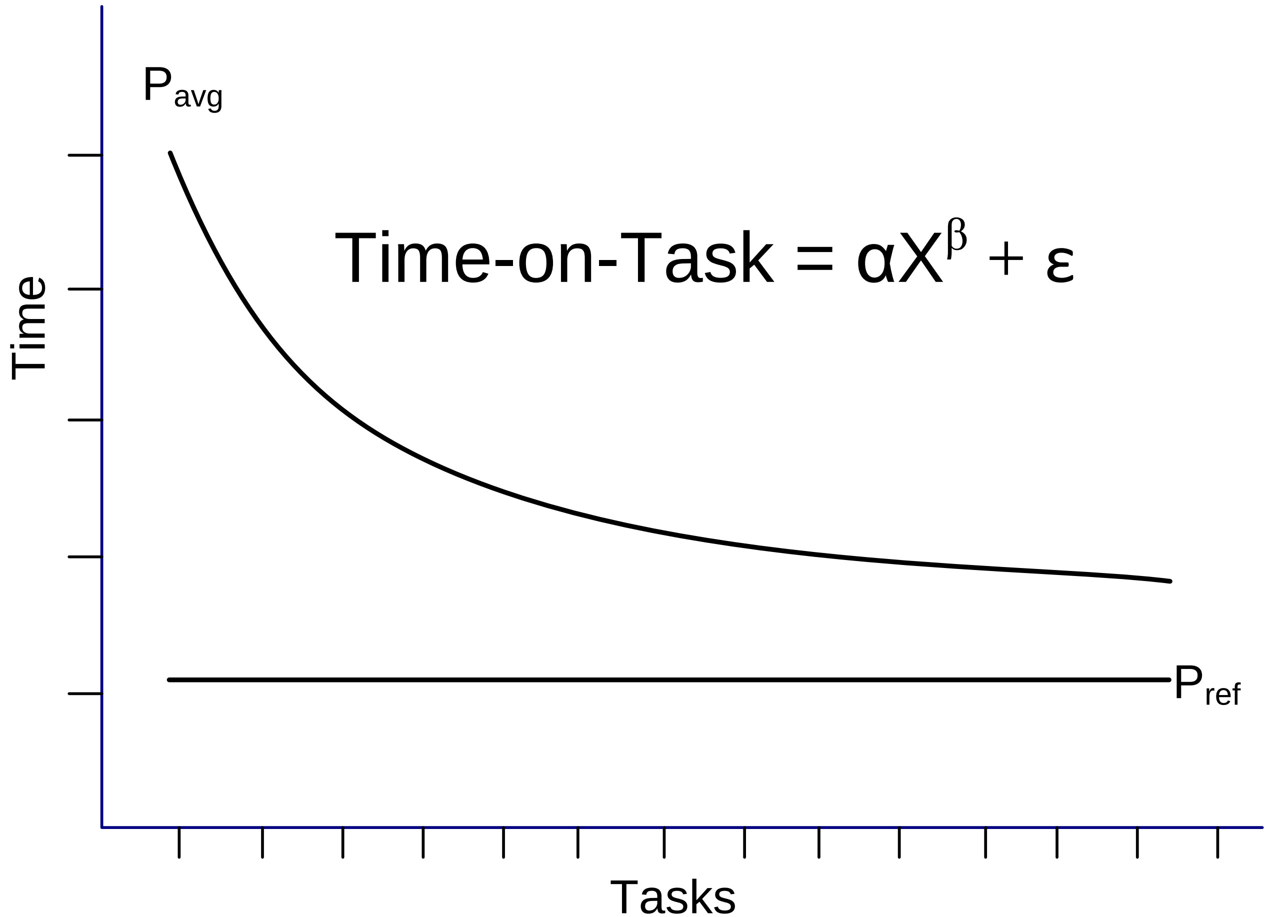
Observations

- 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.

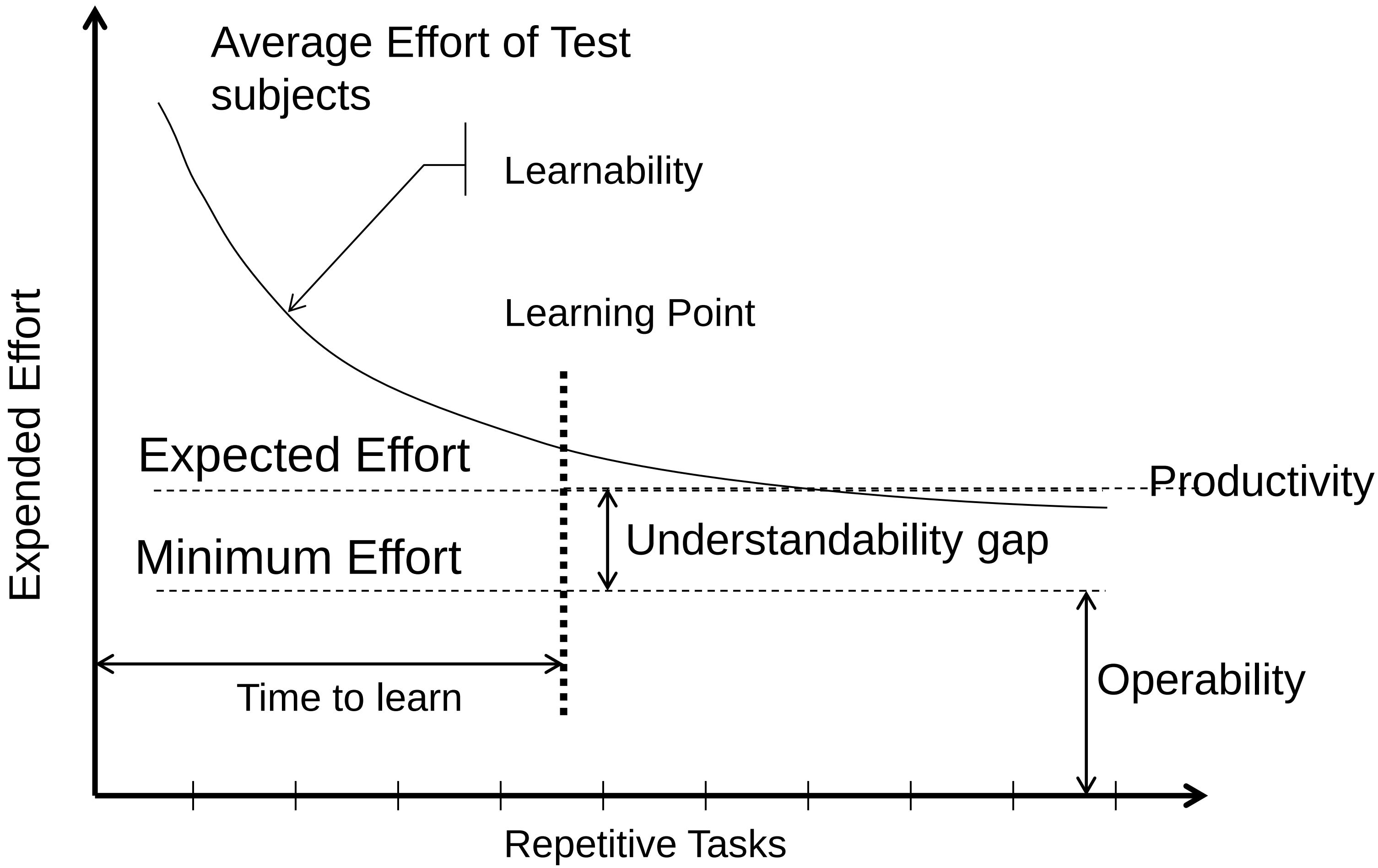
- A set of identical independent (“iid”) experiments on a single scenario can be used to measure learnability and operability

- Eye tracking can be used to provide additional measures of physical and manual effort

Traditional Learning Curve

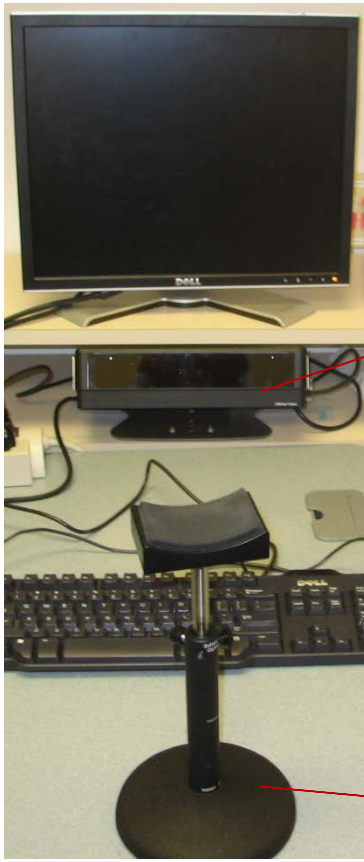


Effort-based Usability Model



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

Eye Tracker Hardware

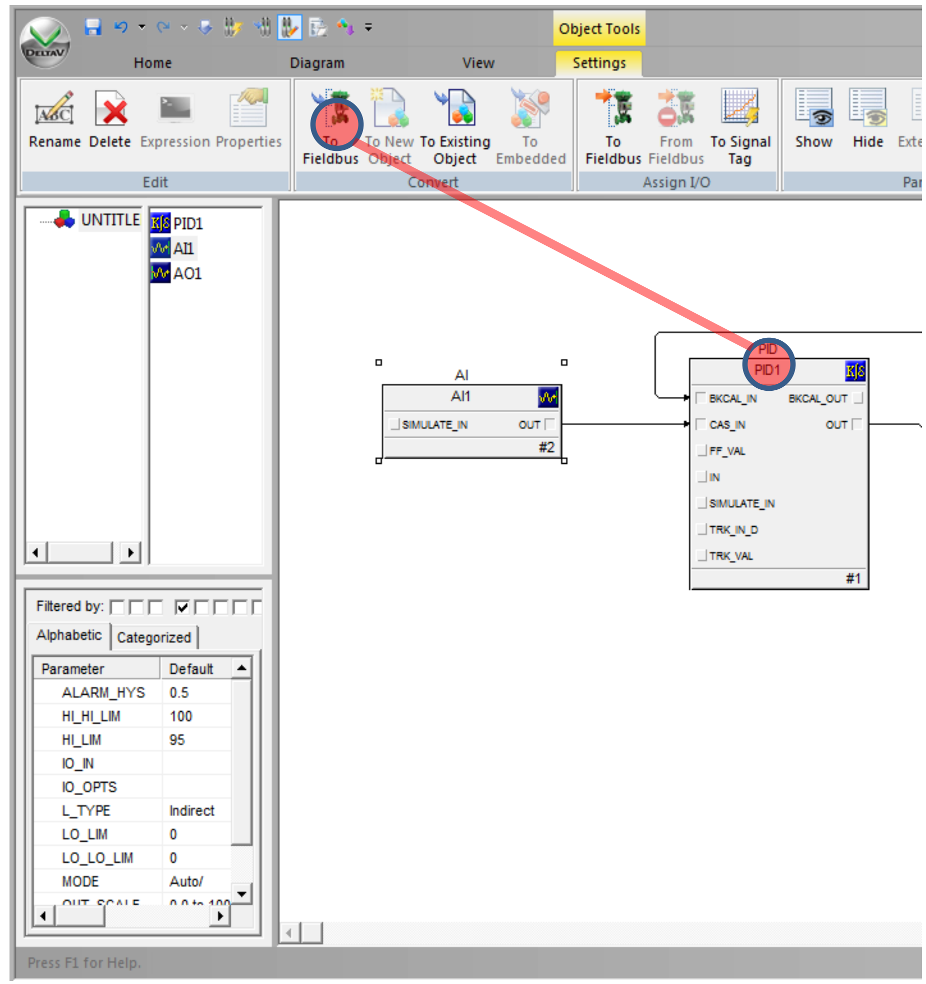


Eye tracker

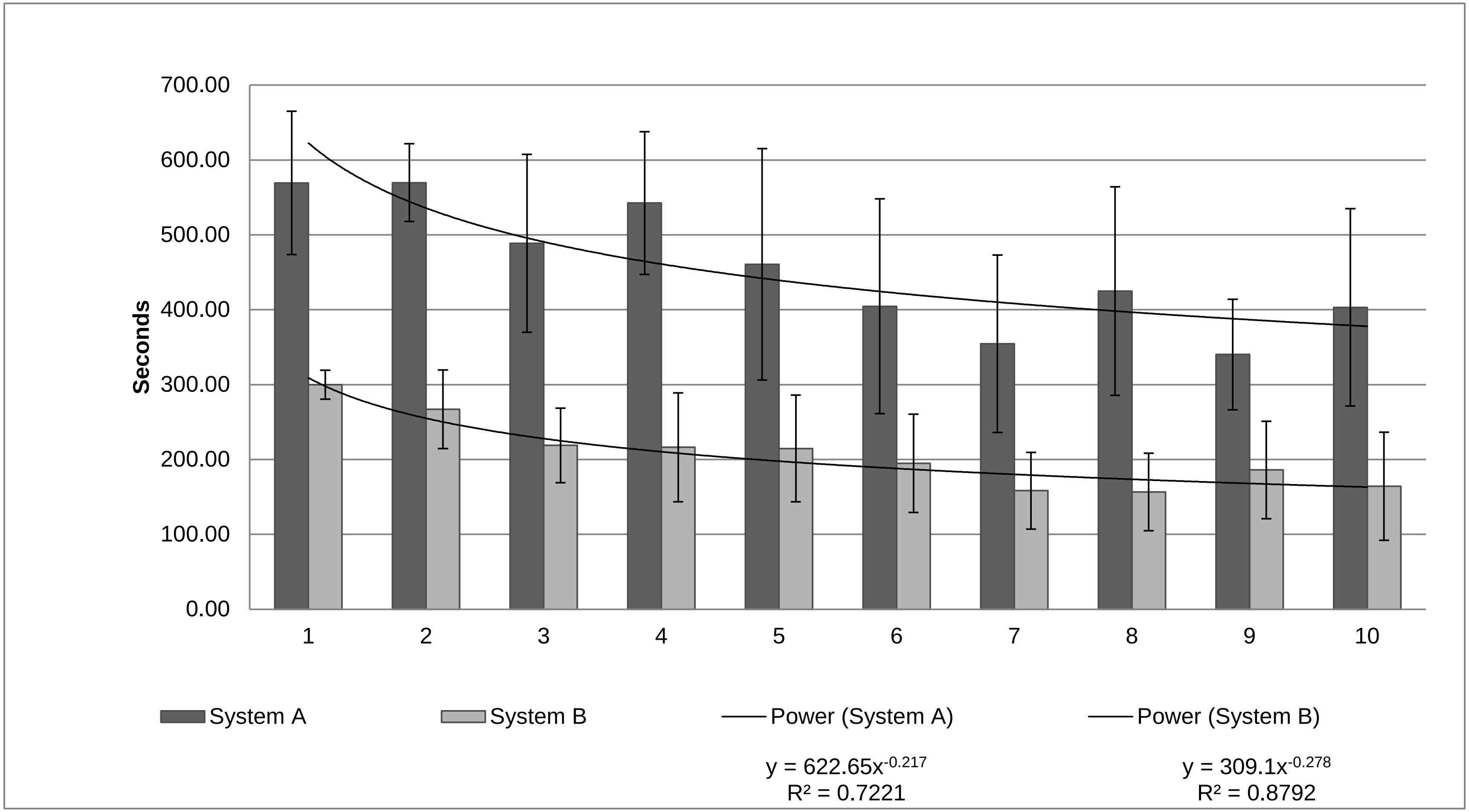
Chin rest

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



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 the 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 – Administrate 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>

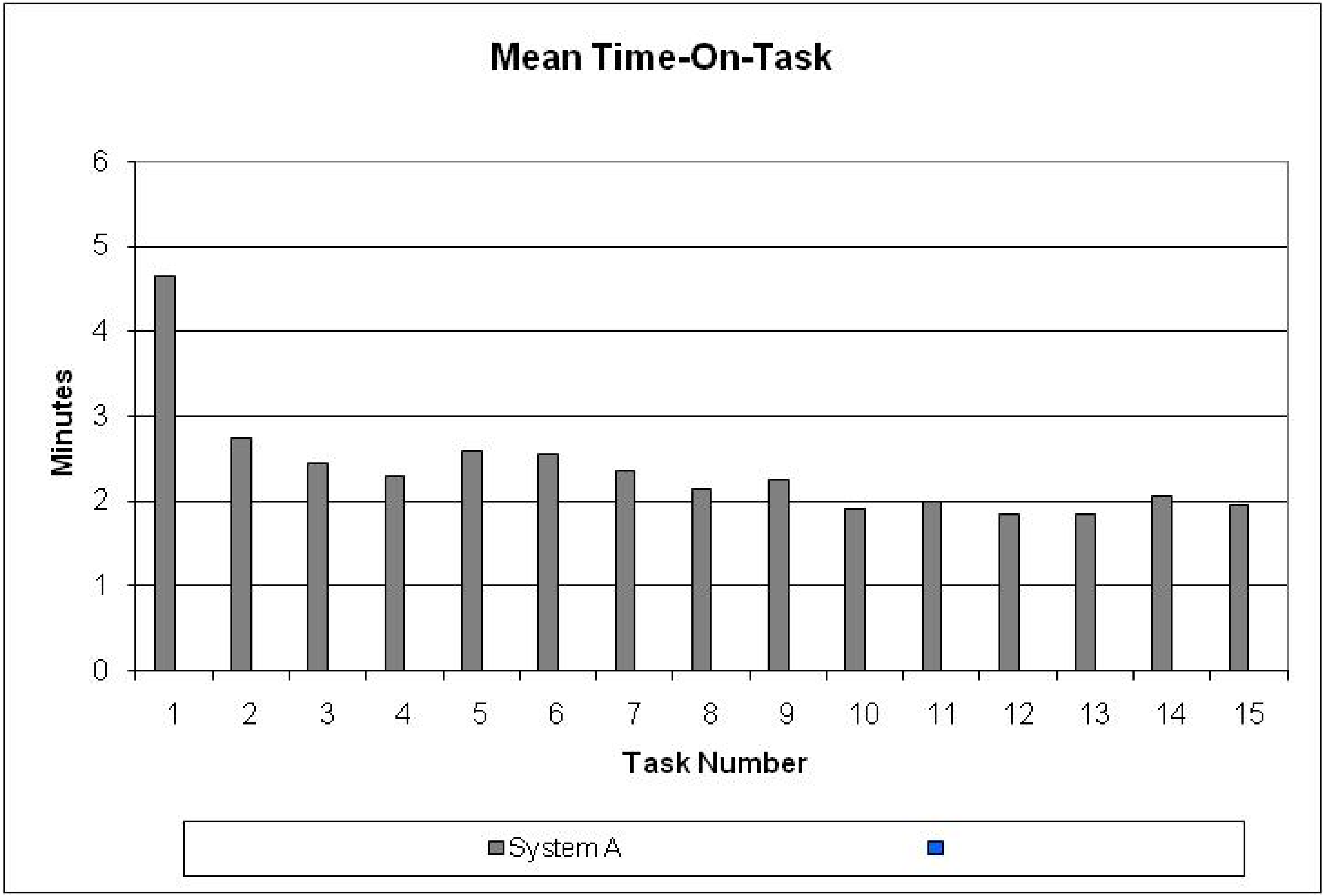
TASK 2

<Start>

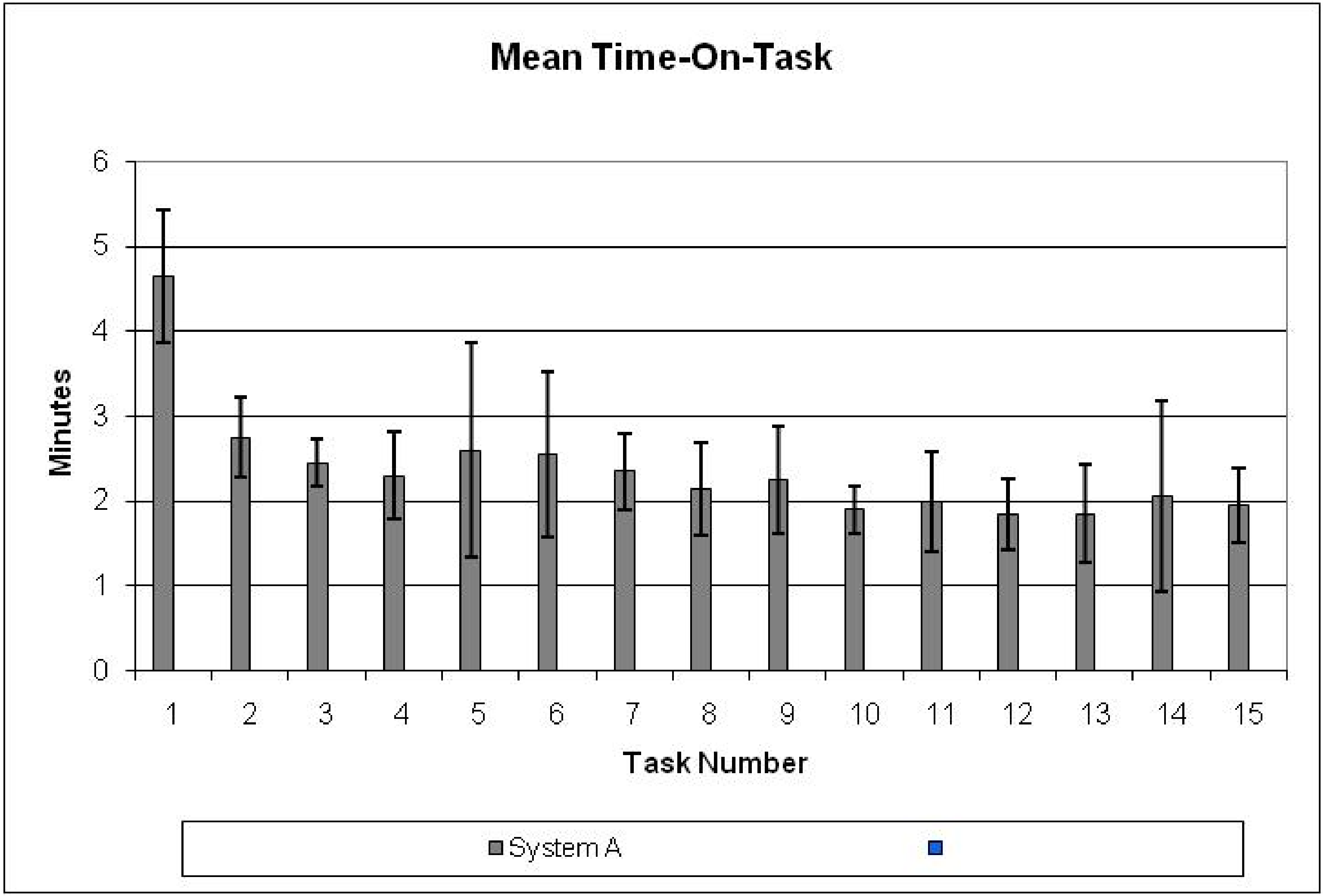
1. Delete block LIC3-16_RSP from the Distillation Column COLUMN1.
2. Add an Analog Input to the left of the block PIC3-15 and name it as PT3-15.
3. Make the following connections -
   a. PT3-15 **OUT** to PIC3-15 **IN** and set the connection as feedback
4. Transfer the changes to the Controller Simulator.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

<End>

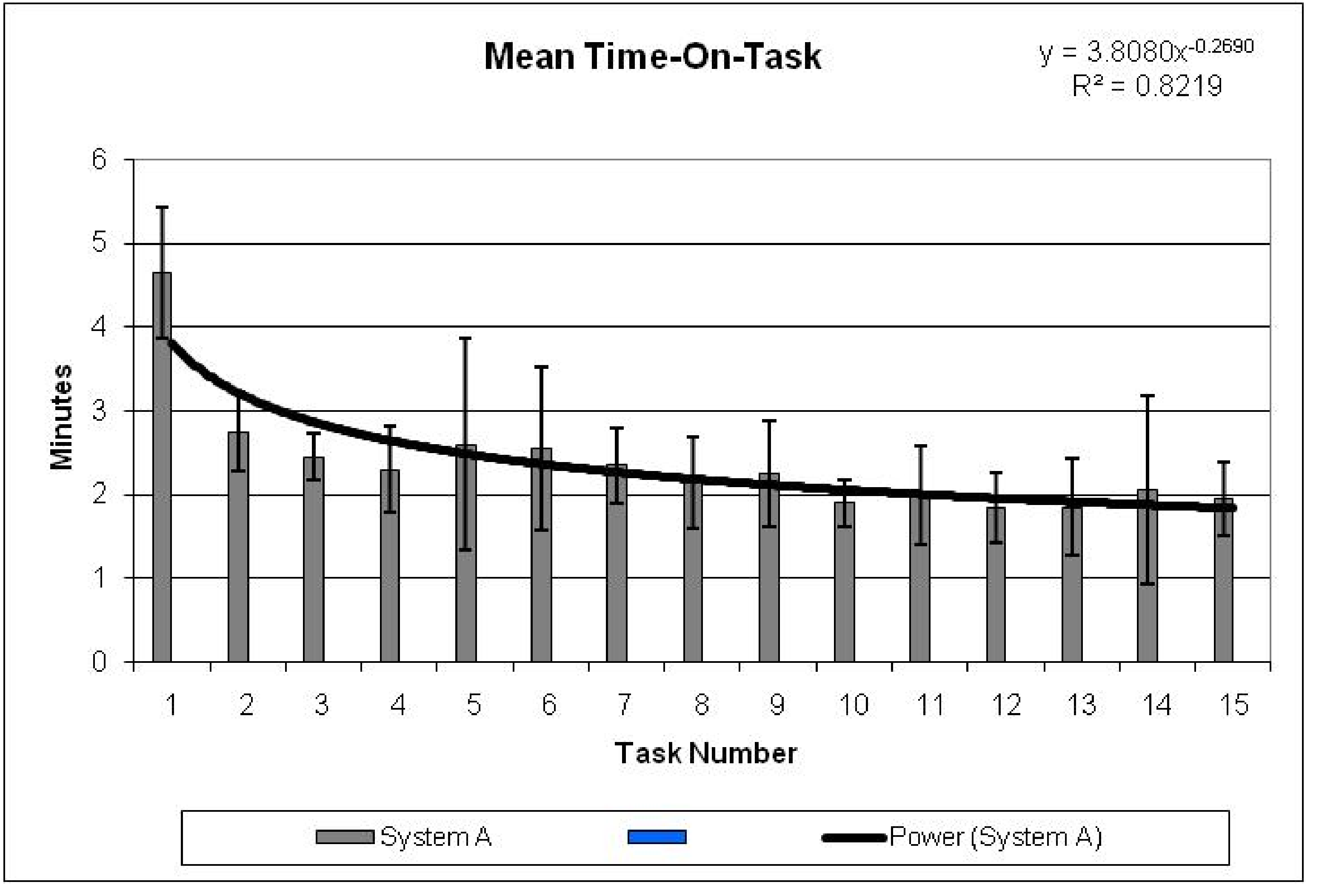
Mean TOT: System A



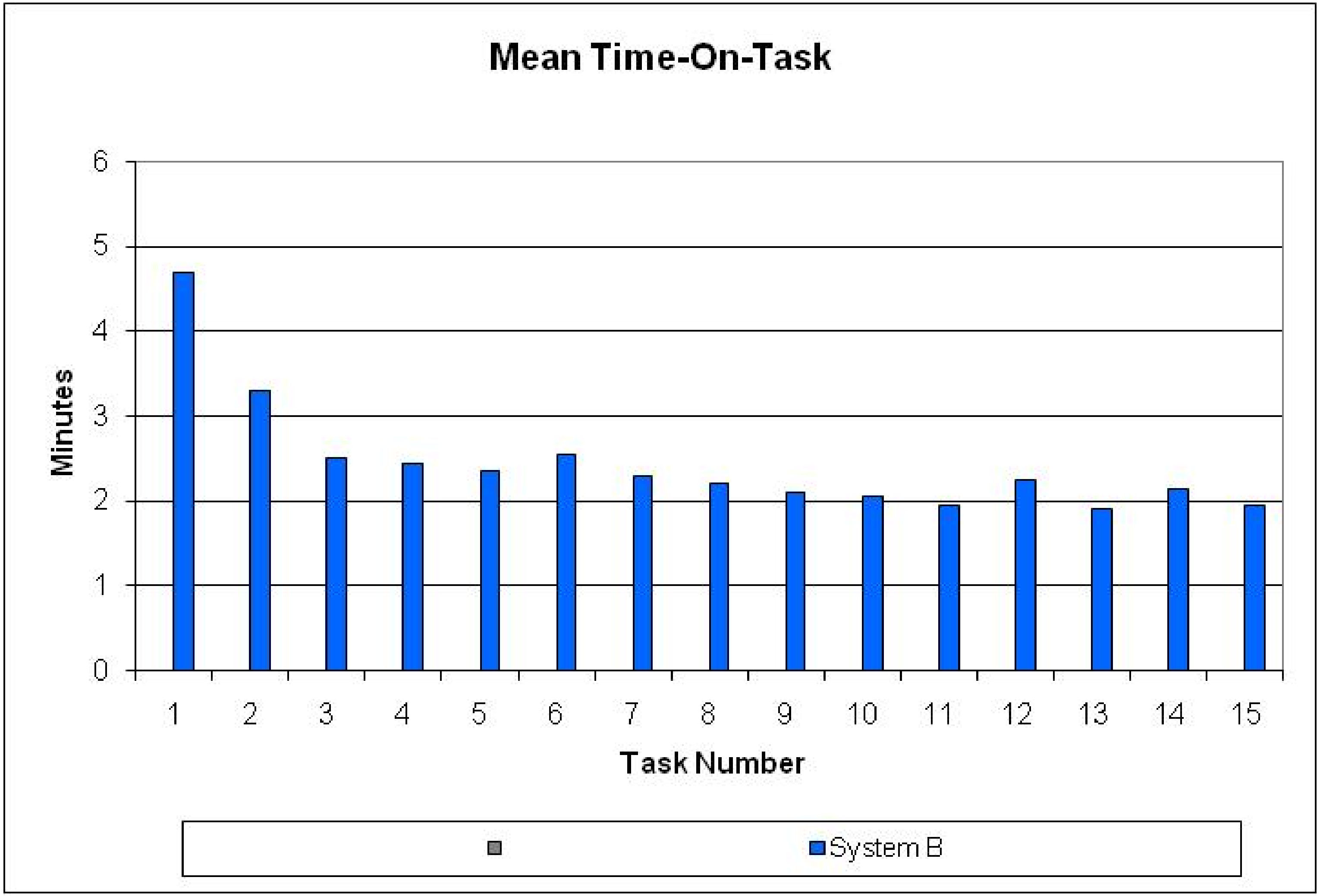
Standard Deviation for ToT in System A



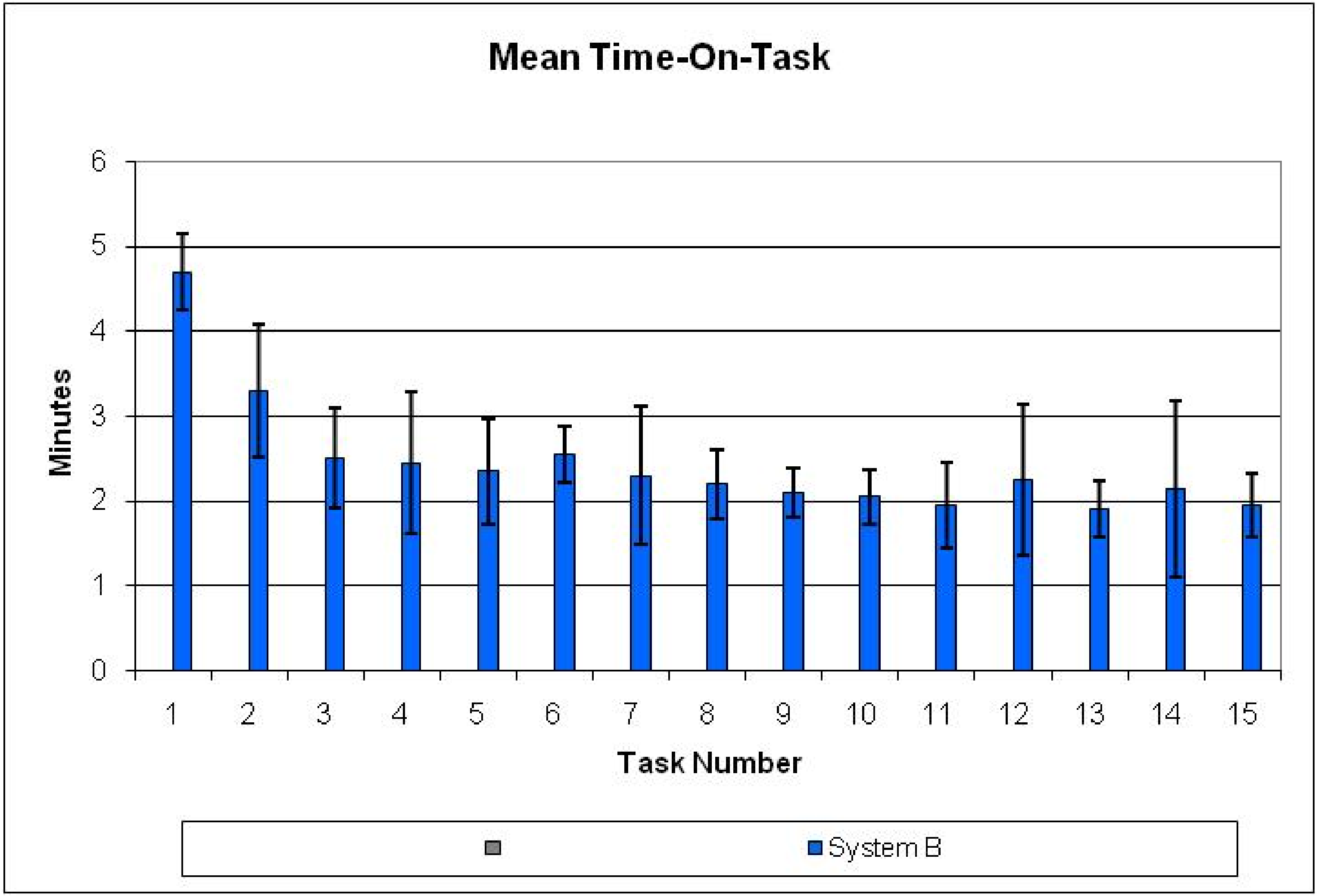
Power Curve Matching to ToT of System A



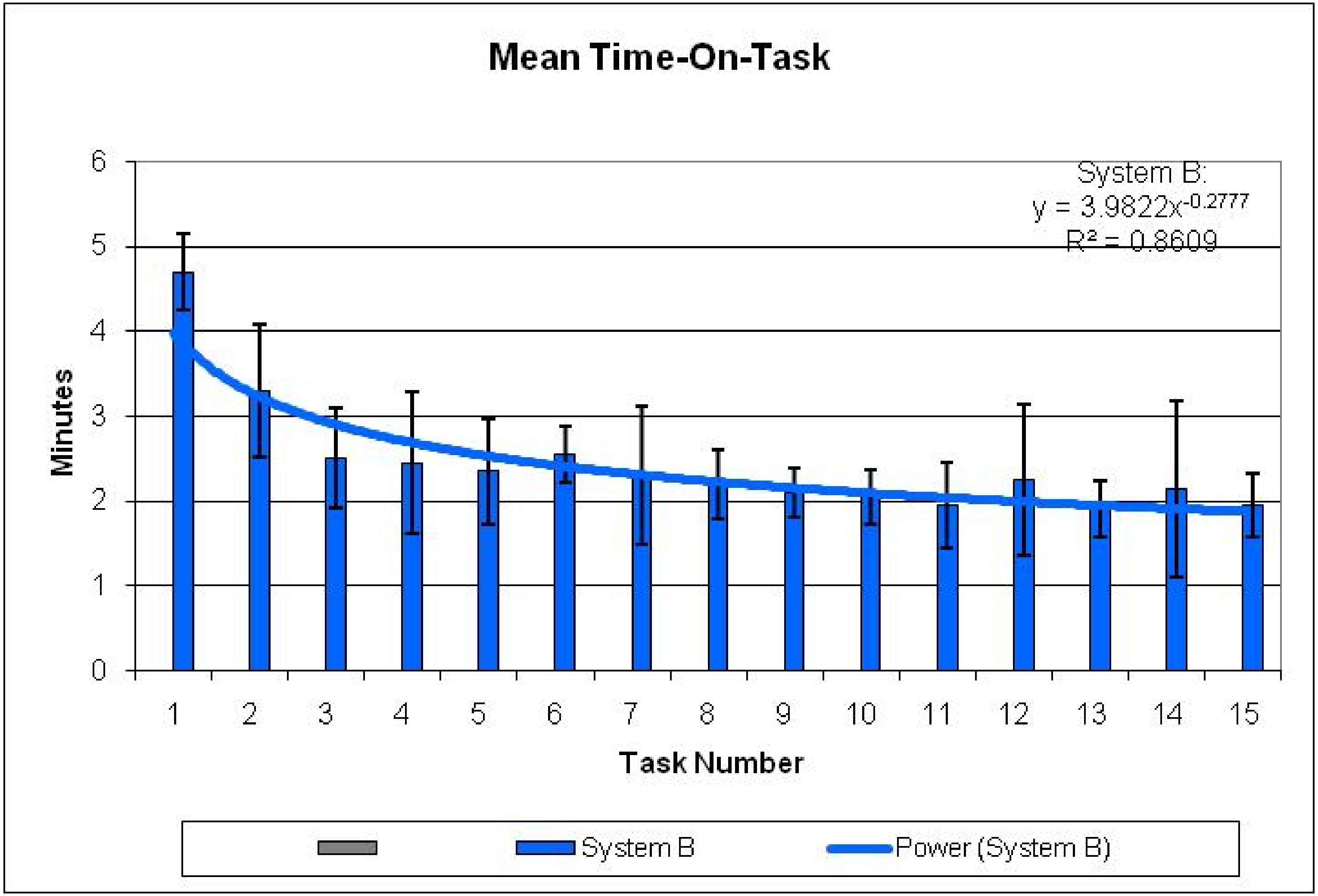
Mean TOT: System B



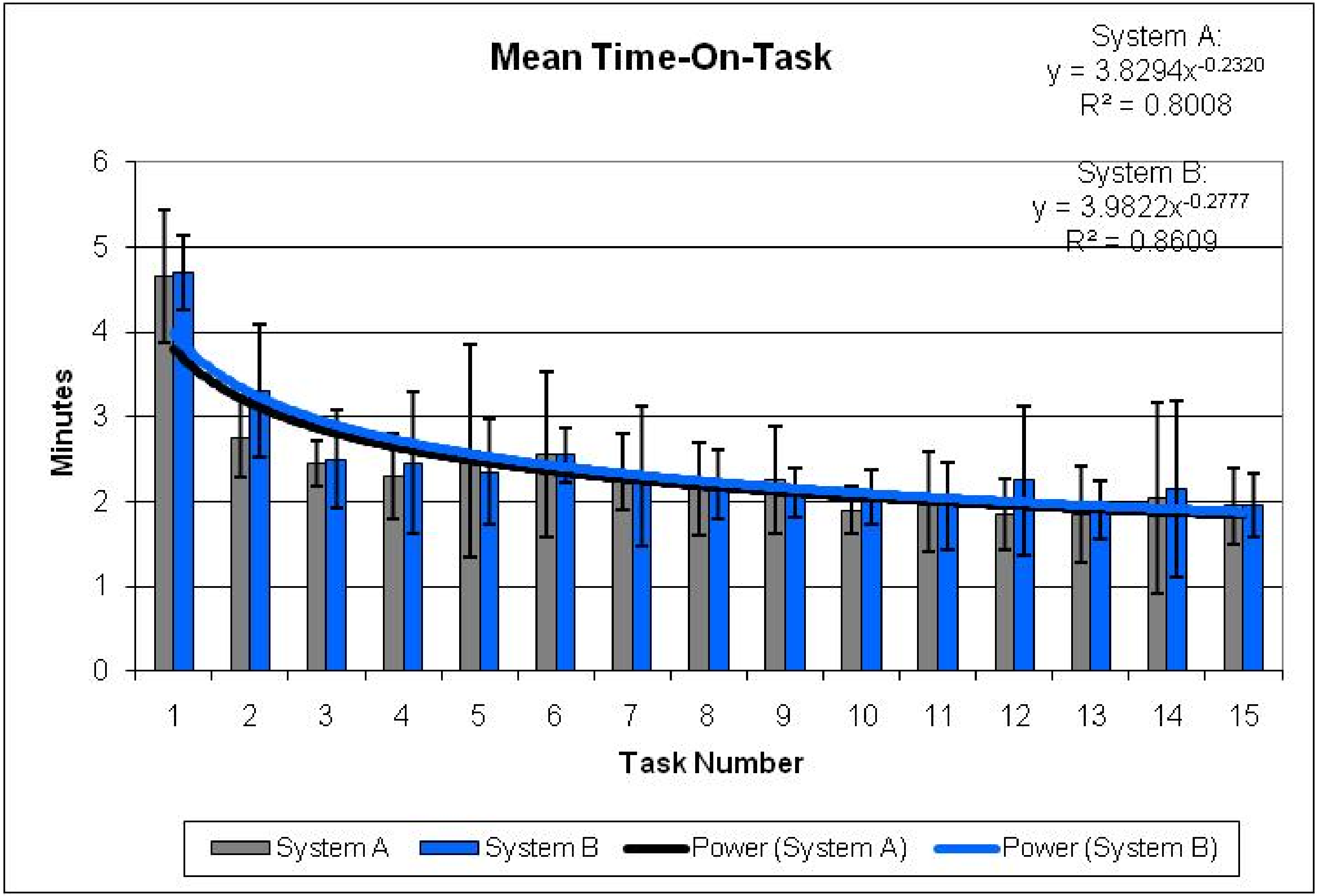
Standard Deviation for ToT in System B



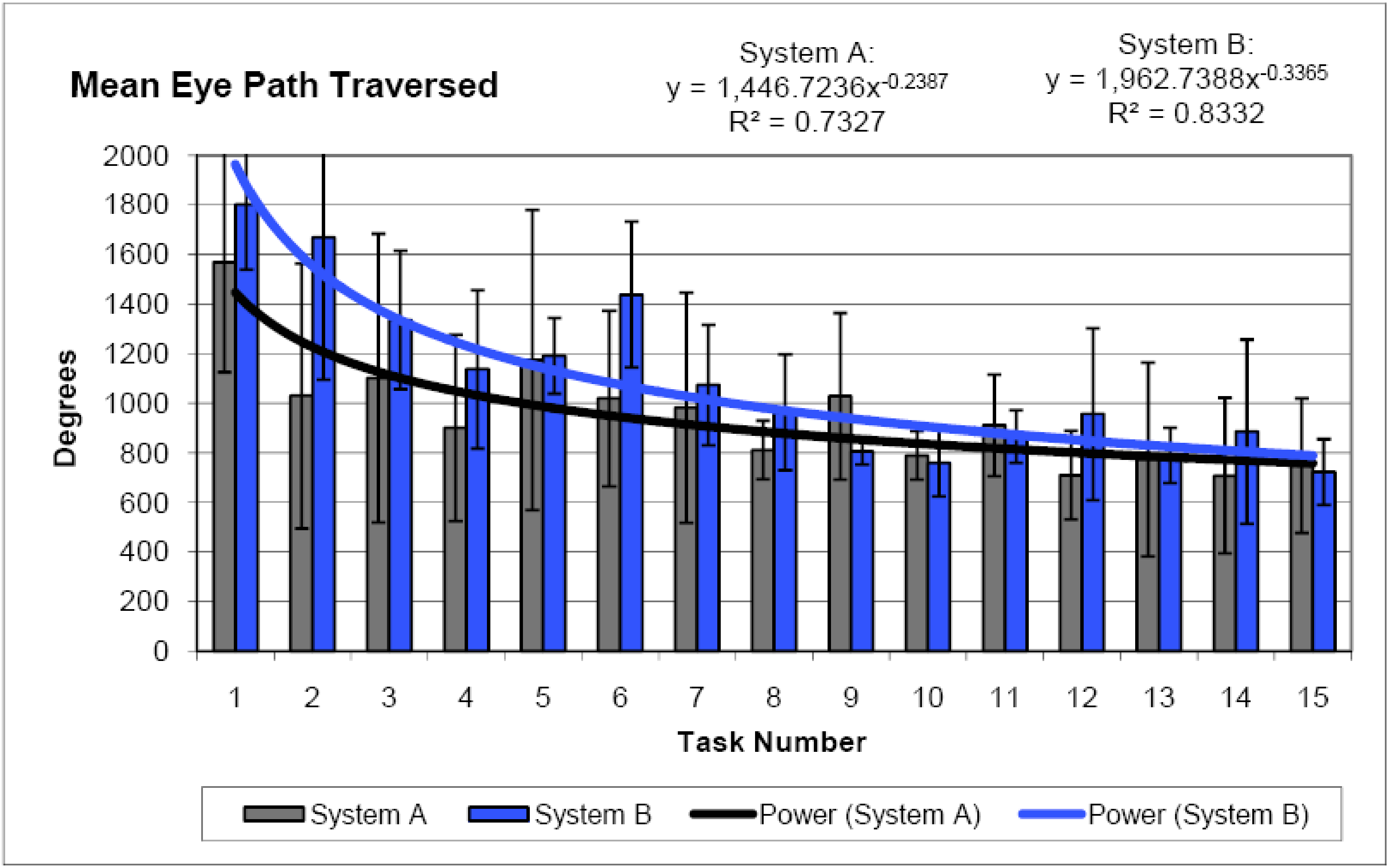
Power Curve Matching to ToT of System B



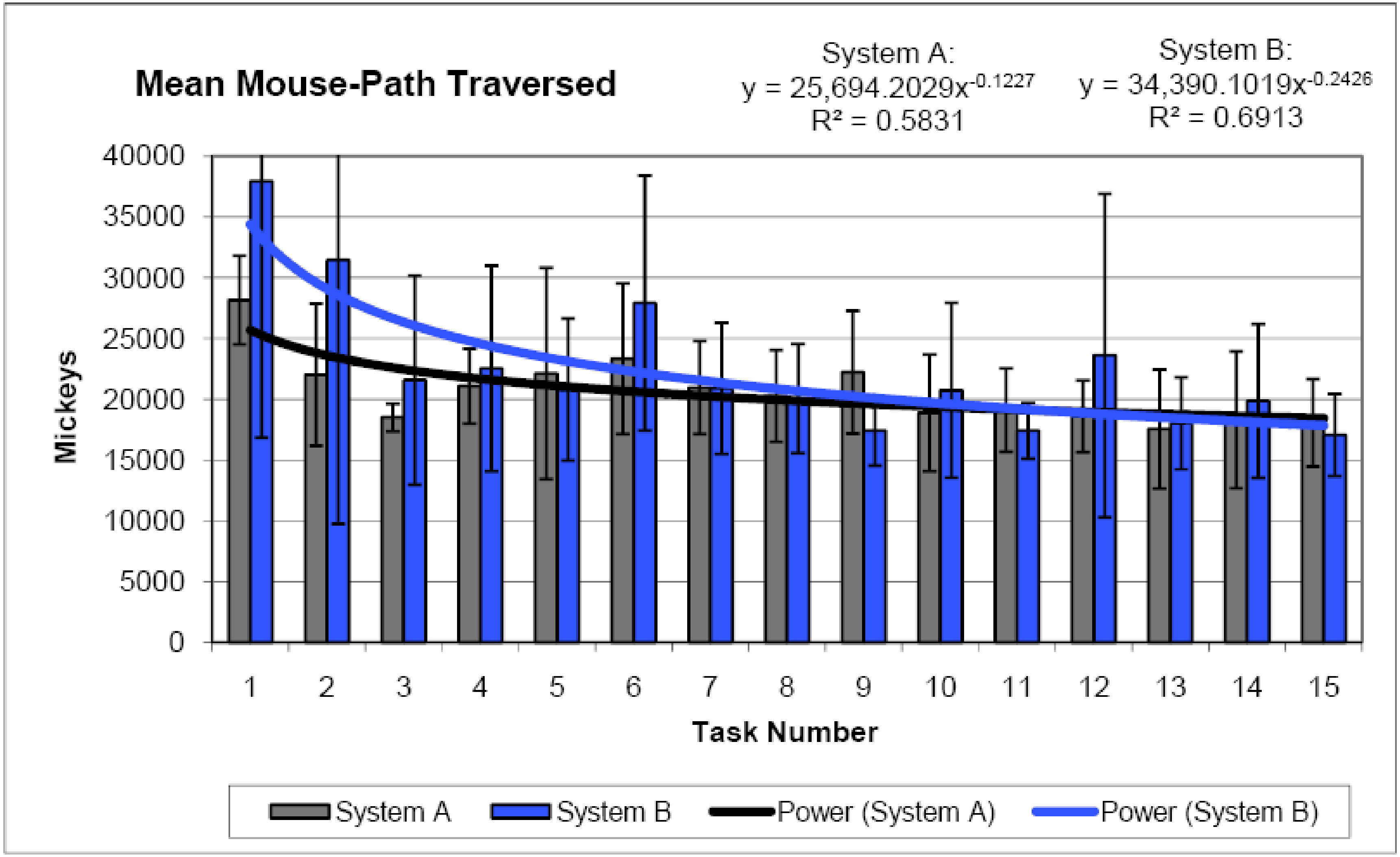
Overall Learnability



Physical Effort



Physical Effort



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

$$R = \begin{bmatrix} r_1 \\ r_2 \\ . \\ . \\ r_j \\ \\ \end{bmatrix}$$

$r_1 =$ Average Saccade Amplitude

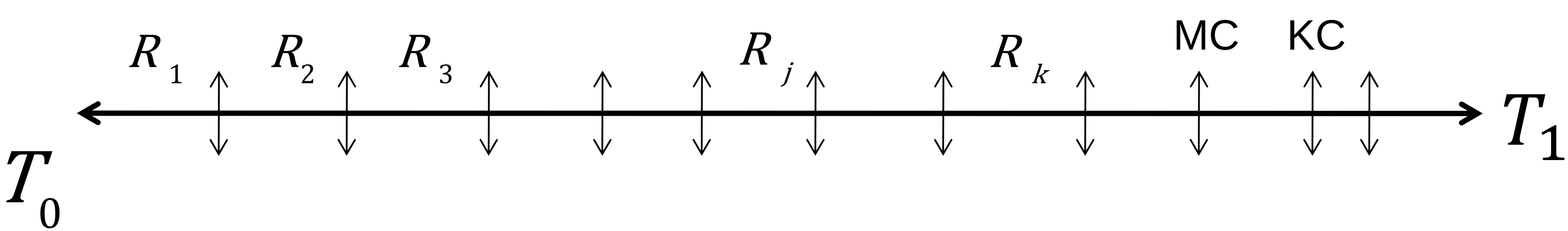
$r_2 =$ Average Fixation duration

$r_j =$ Total Time on Task

$$T_0 \longleftrightarrow T_1$$

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

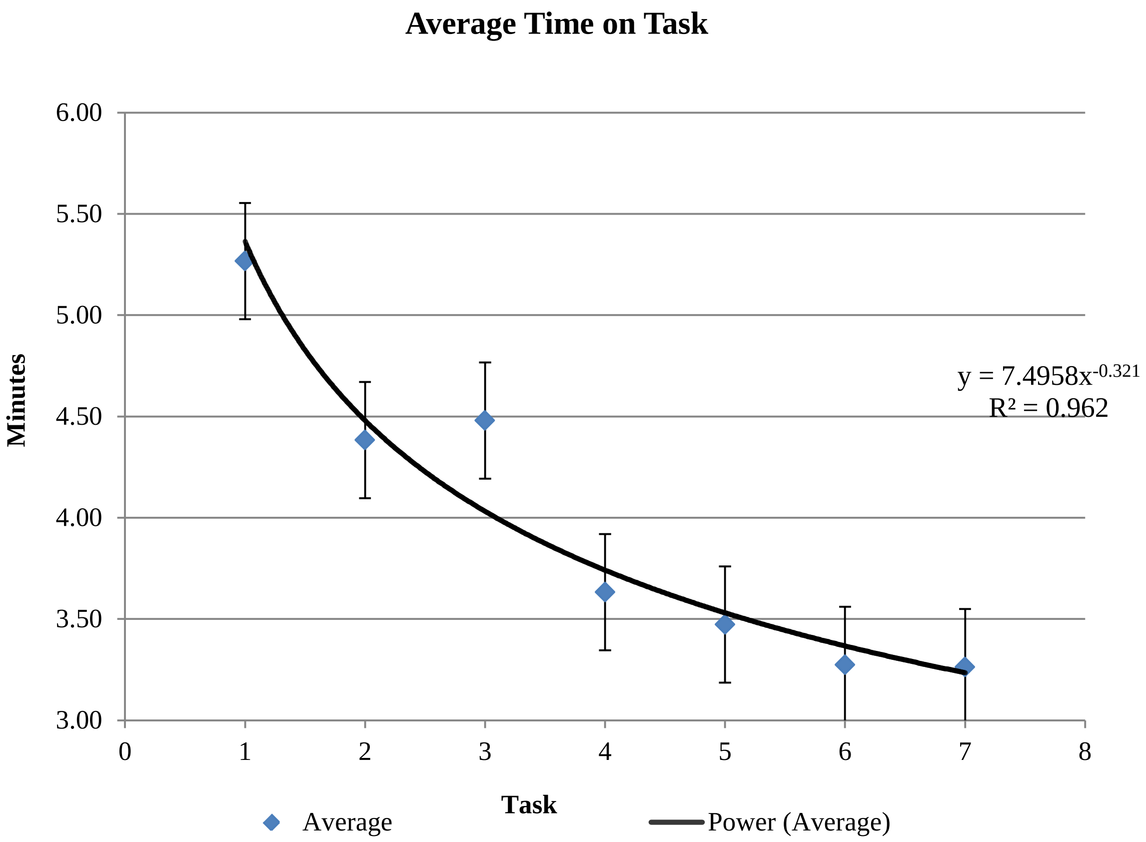
- **Pinpoint Analysis:** Identifying and pinpointing issues with the interface.

- ***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.



Pattern Recognition

- Assignment of *label*s 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 O**perations**

The following are various relevant pattern recognition operations

- Segmentation
- Feature Extraction and Feature Selection
- Principal Component Analysis (PCA)
- Clustering
- Applying a Threshold

Pattern Recognition Operations (Continued)

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.

Pattern Recognition Operations (Continued)

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

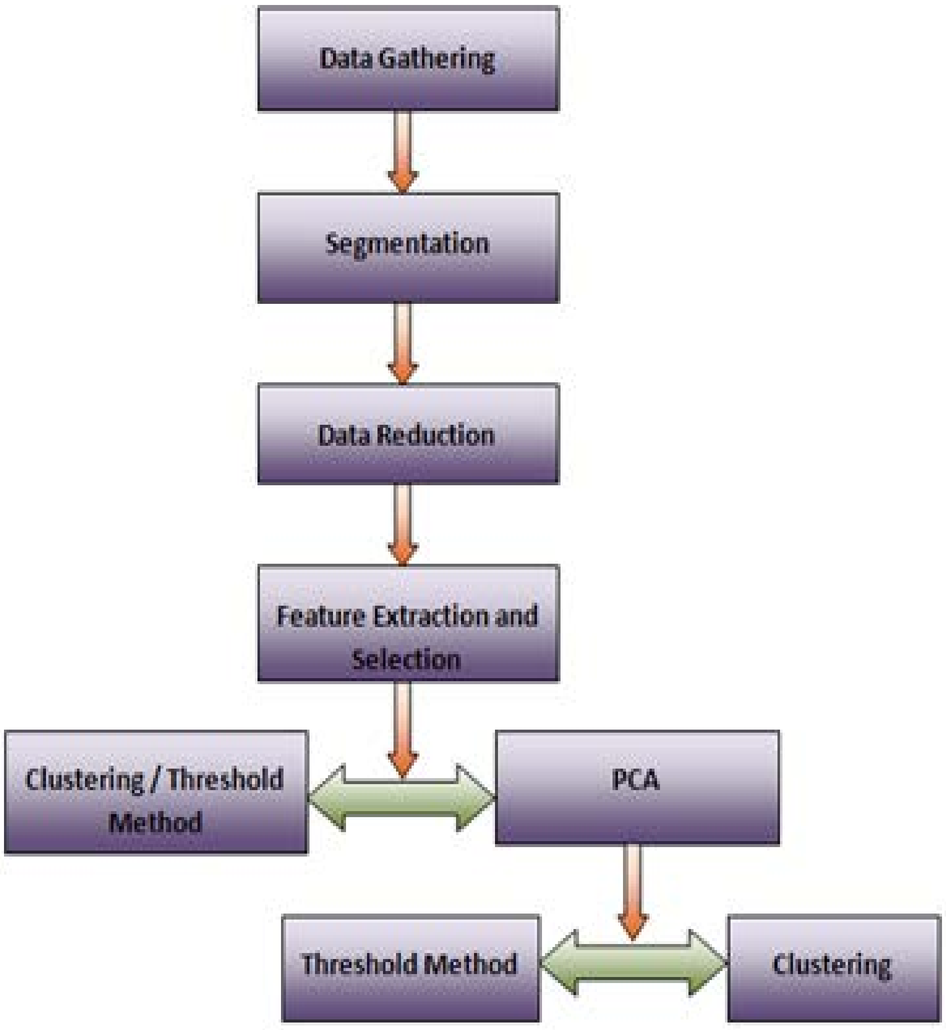
Difference between Thresholding and Clustering:

- A threshold is applied only on individual features or linear combination of features.
- Clustering is applied on multi-dimensional data.

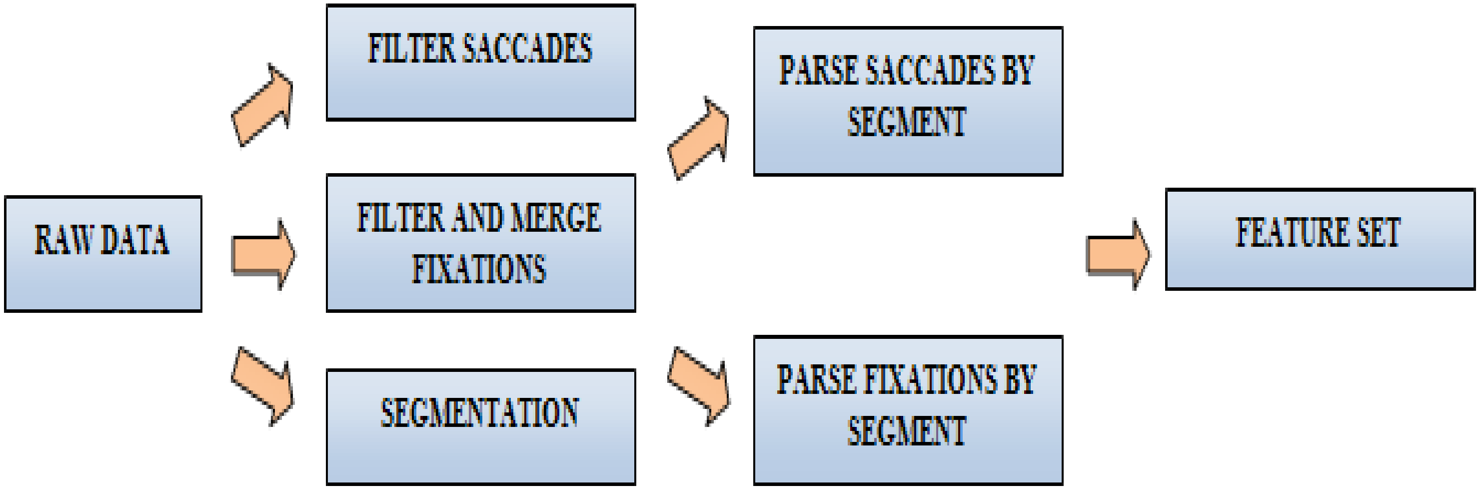
Experiment Test Procedure

The experiment procedure includes the following three main phases

- Phase 1 - Data Gathering
- Phase 2 – Reduction (includes segmen 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

Experiment Test Procedure (Continued)

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 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****; s*egments 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

Number of Fixations			
Segment Start Time	Segment End Time	Manual Classification	Tool Classification
0	551	NE	NE
551	1451	NE	E
1451	3088	NE	E
3088	5640	NE	E
5640	5640	NE	NE
5640	10880	E	E
10880	11296	NE	NE
11296	11488	NE	NE
11488	11681	NE	NE
11681	11921	NE	NE
11921	17840	NE	NE
17840	17840	NE	NE
17840	20921	NE	E
20921	22670	A	E
22670	22670	A	NE
22670	28409	A	E
28409	30090	A	E
30090	31731	A	E
31731	33722	A	E
33722	37232	A	E
37232	37584	A	NE
37584	37728	A	NE
37728	37904	A	NE
37904	38416	A	NE
38416	40892	NE	NE

Type-I Errors, Type-II Errors, and Inspection Time

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

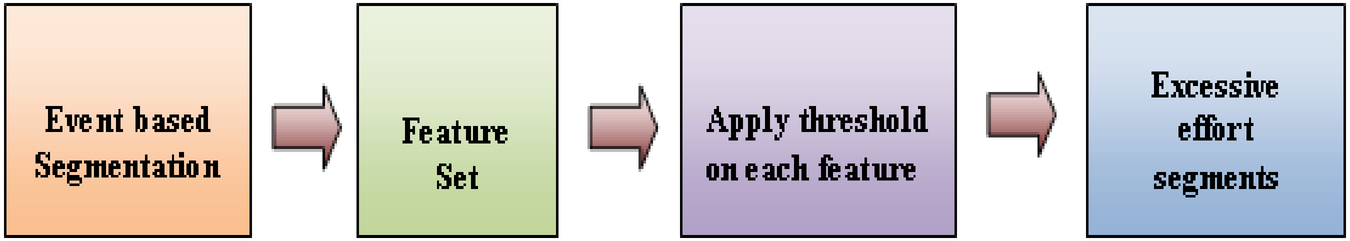
Experiment 2: Identifying excessive effort segment using K-means clustering

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

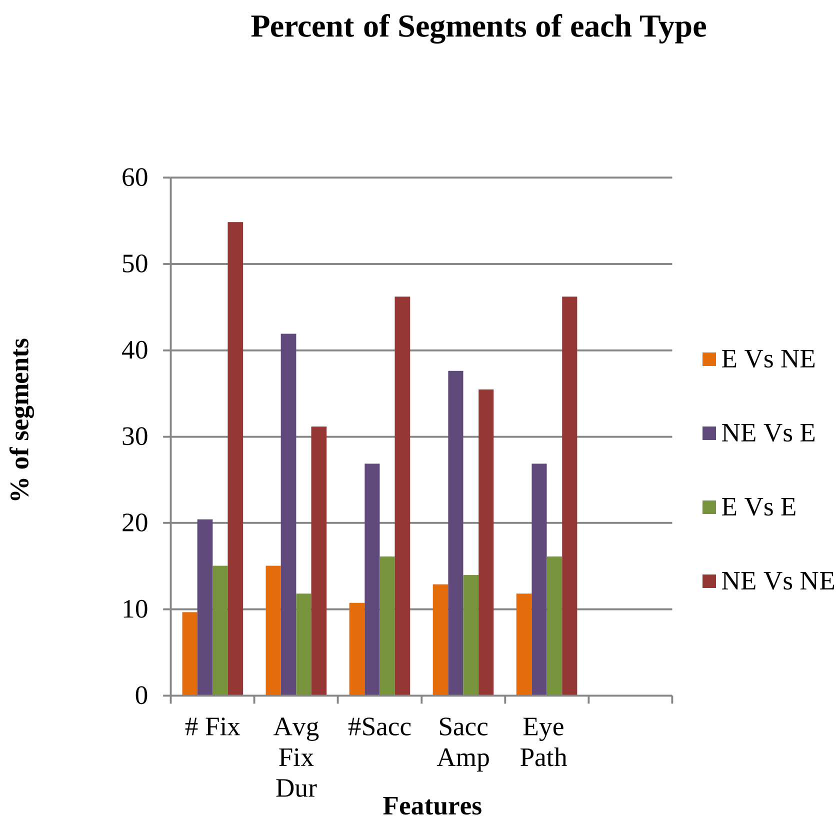
Identifying excessive effort segments using the threshold method

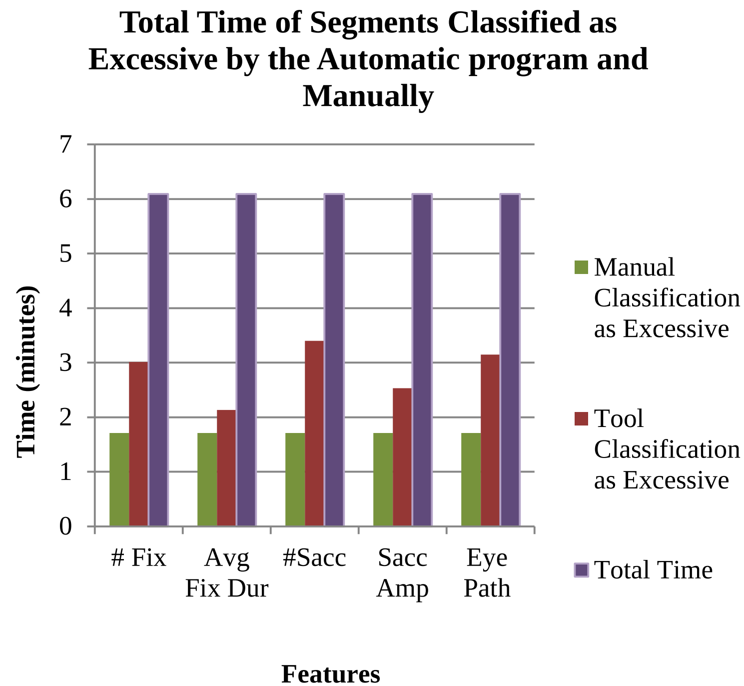


Feature set

- Number of fixations
- Average fixation duration
- Number of saccades
- Average saccade amplitude
- Eye path traversed.

Data File 1





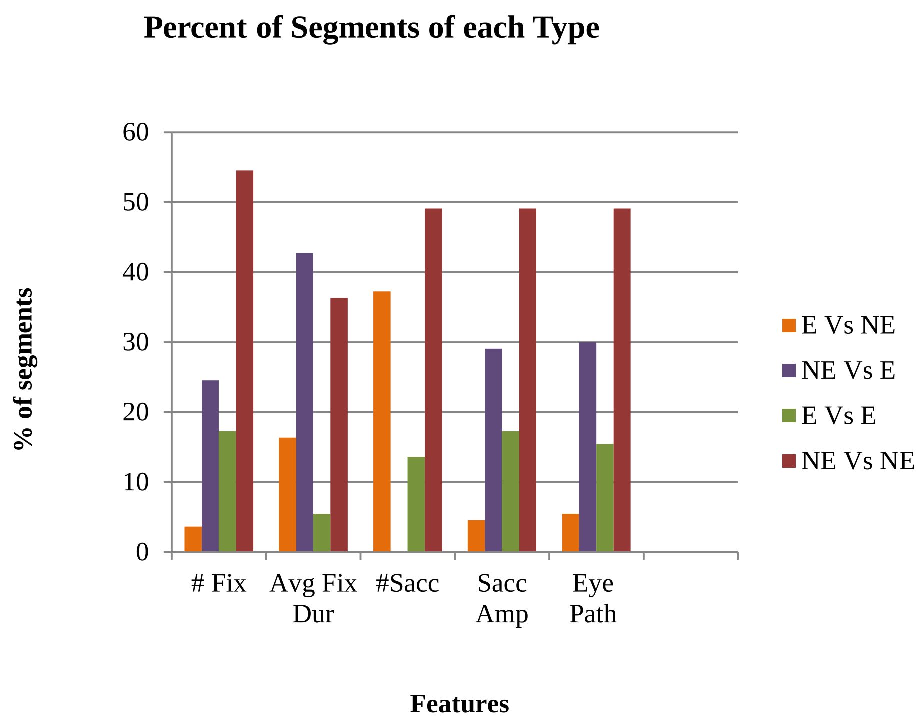
Observation:

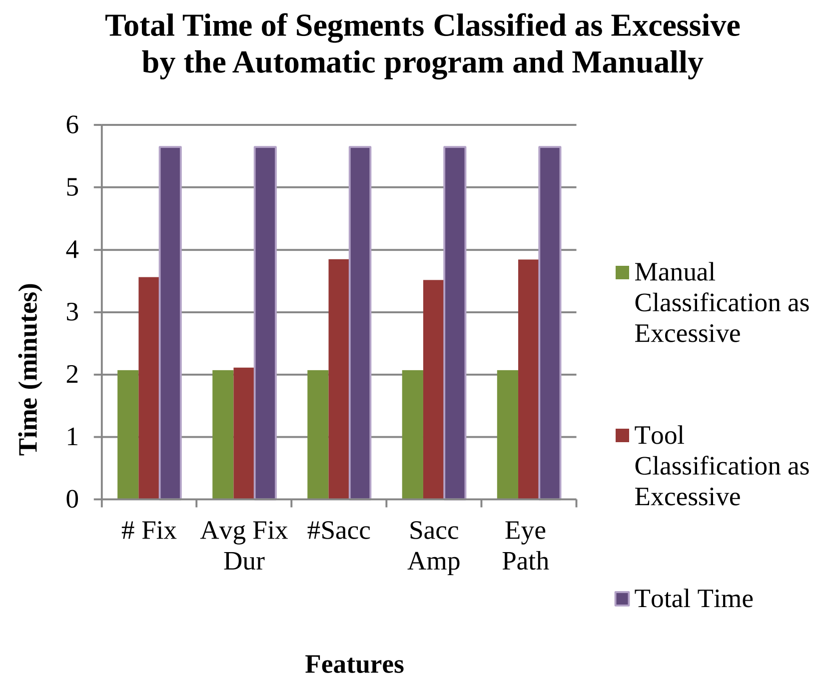
- ‘number of fixations’ performs well in terms of type-II errors.

Observation:

- ‘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:

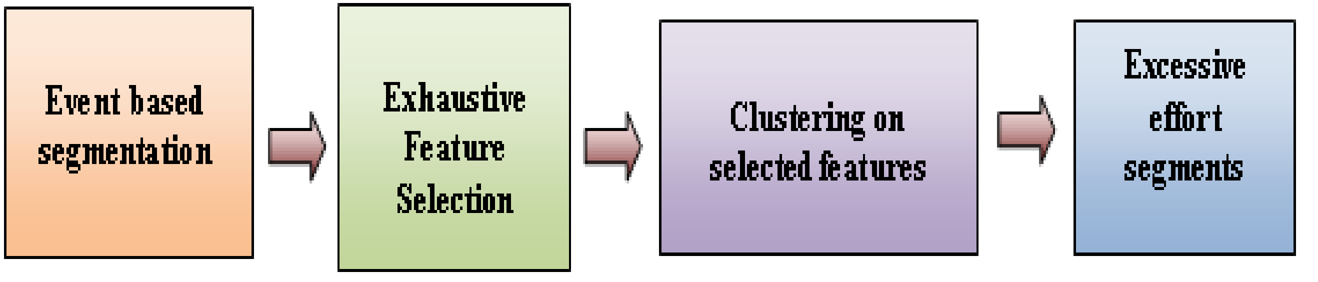
- ‘number of fixations’ performs well in terms of type-II errors.

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

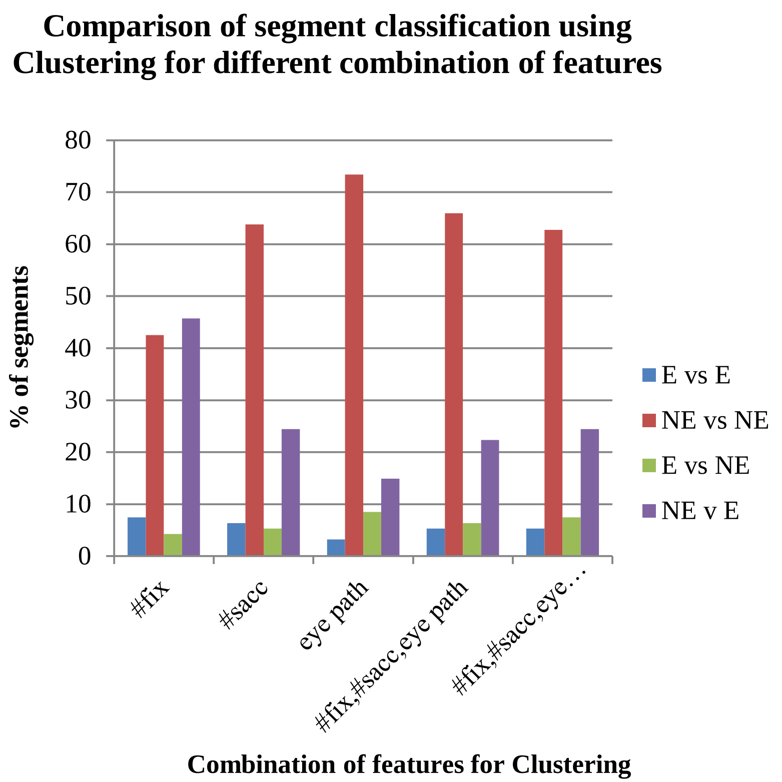
Identifying excessive effort segments using exhaustive feature selection and K-means clustering.



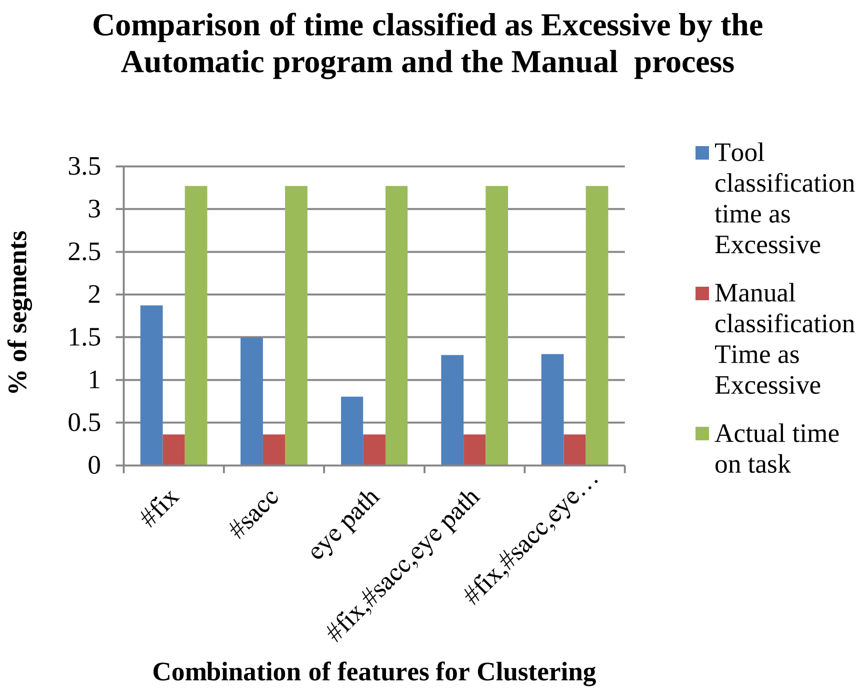
Feature set

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

Data File 2



Observation: feature subset 1 performs well in terms of type-II errors.

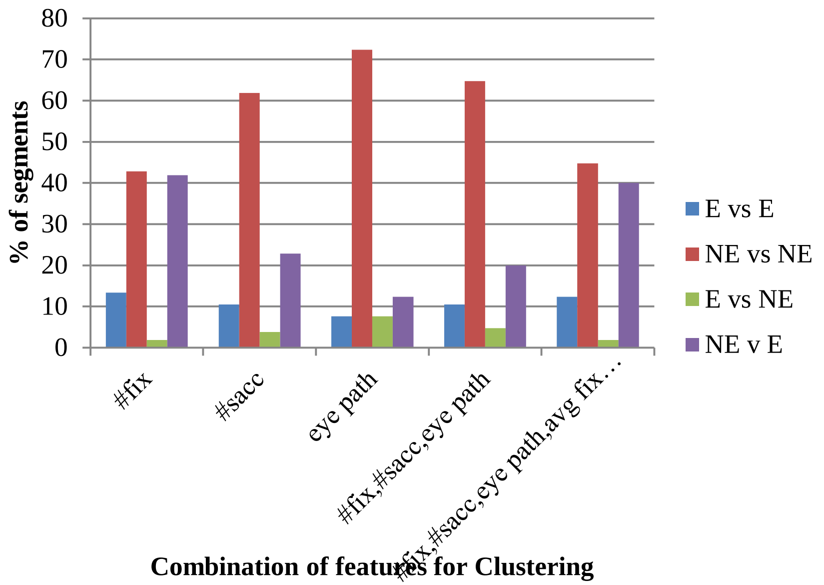


Observation:

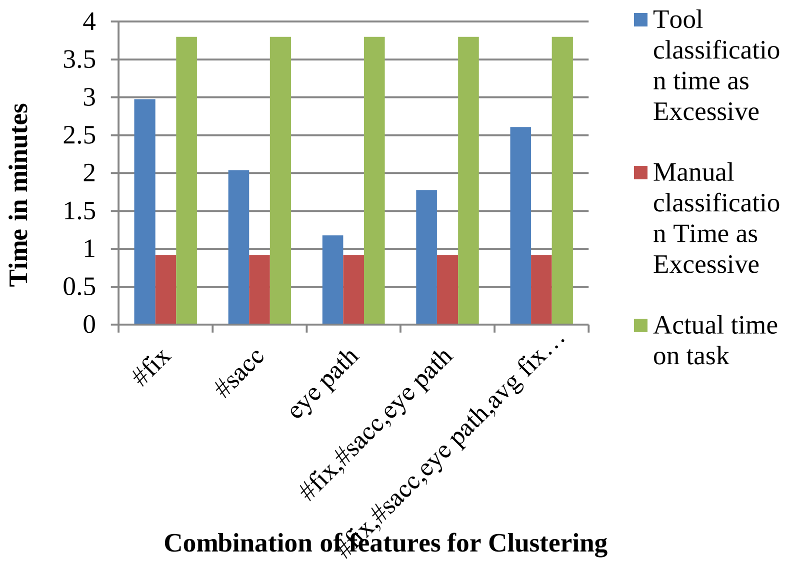
Subset 3, with 'eye path traversed' as a feature value has very small inspection time and type-II errors in an acceptable range.

Data File 3

Comparison of segment classification using Clustering for different combination of features



Comparison of time classified as Excessive by the Automatic program and the Manual process



Observation:

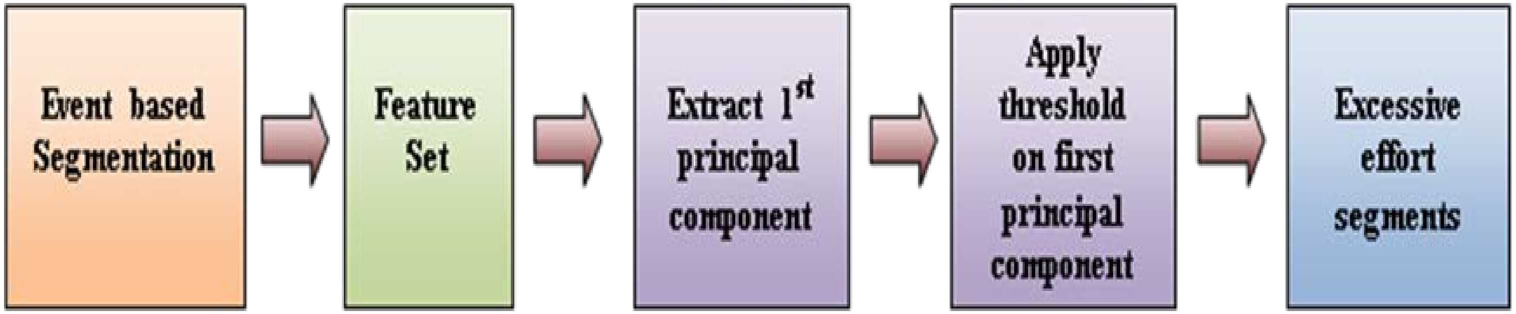
- feature subset 1 and subset 5 have minimum type-II errors

Observation:

- 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

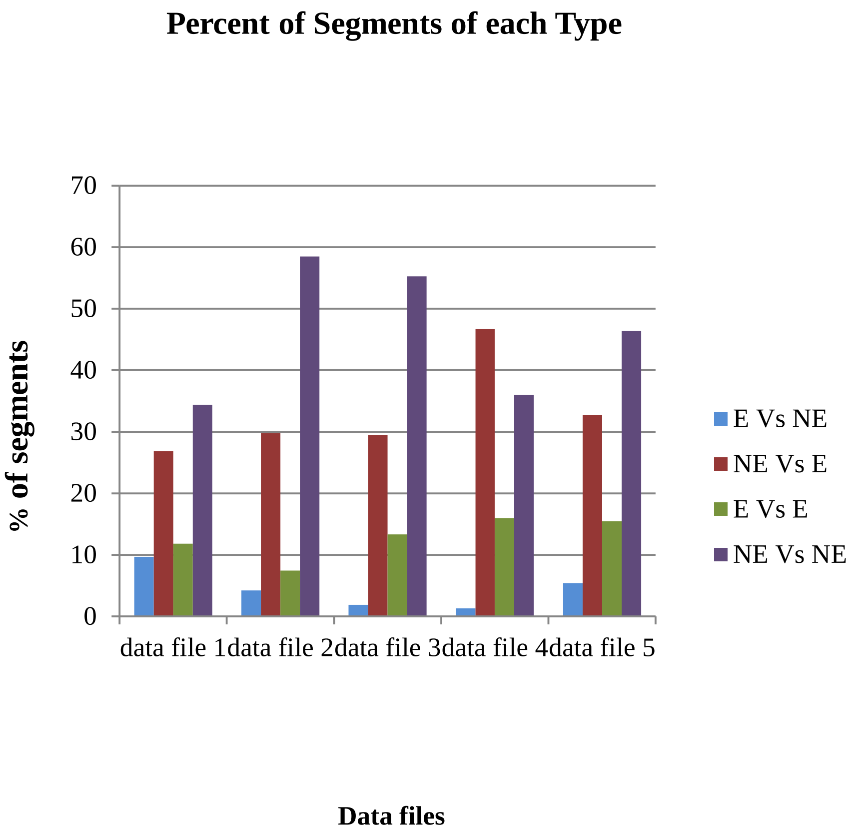
Identifying excessive effort segments using principal component analysis and thresholding

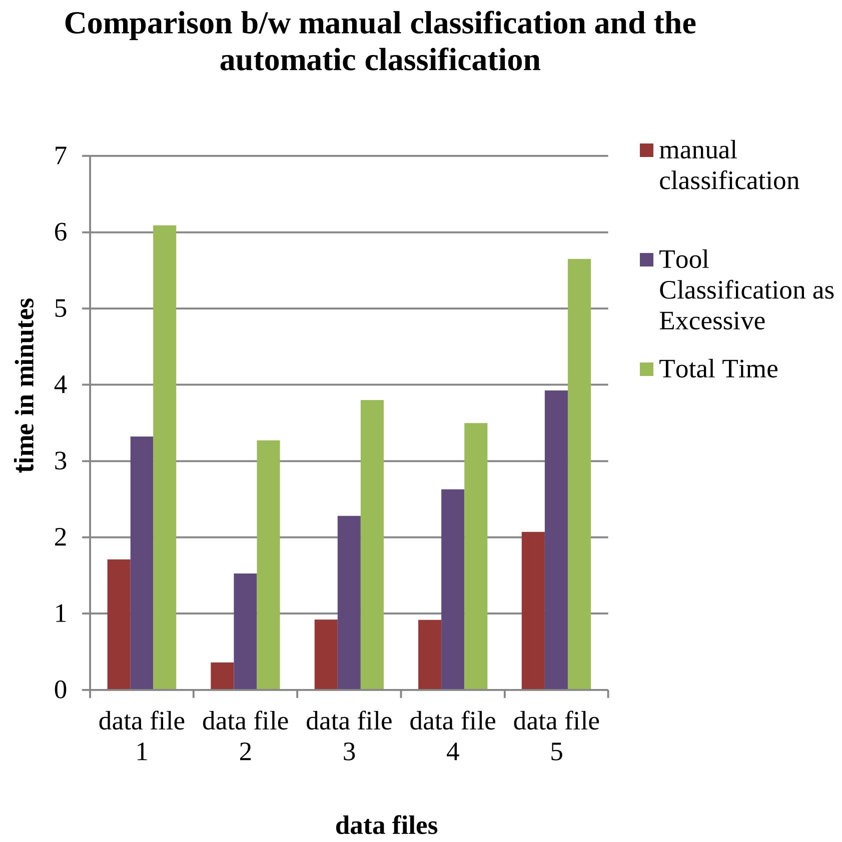


Feature set:

1st principal component

Data File Analysis





Observation:

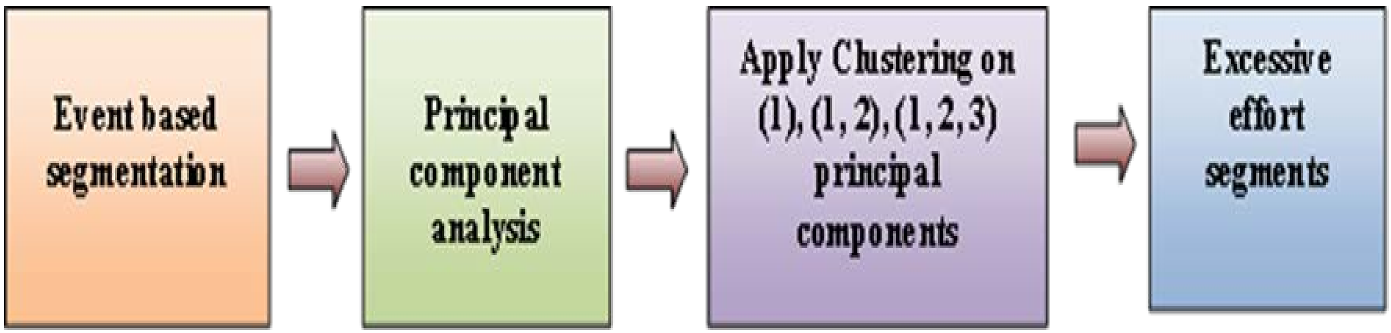
➢ relatively low values for type-II errors for all data files.

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



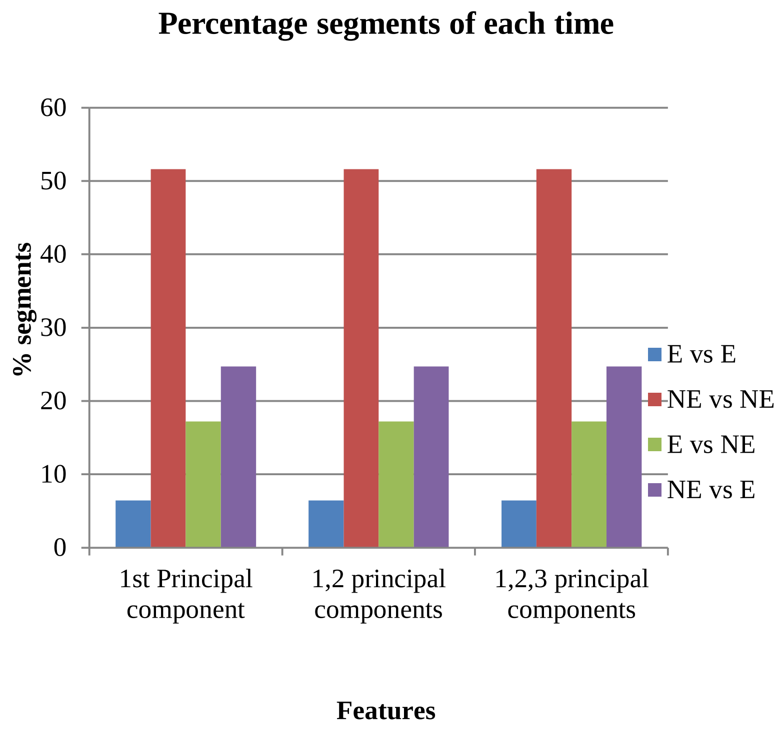
Feature set:

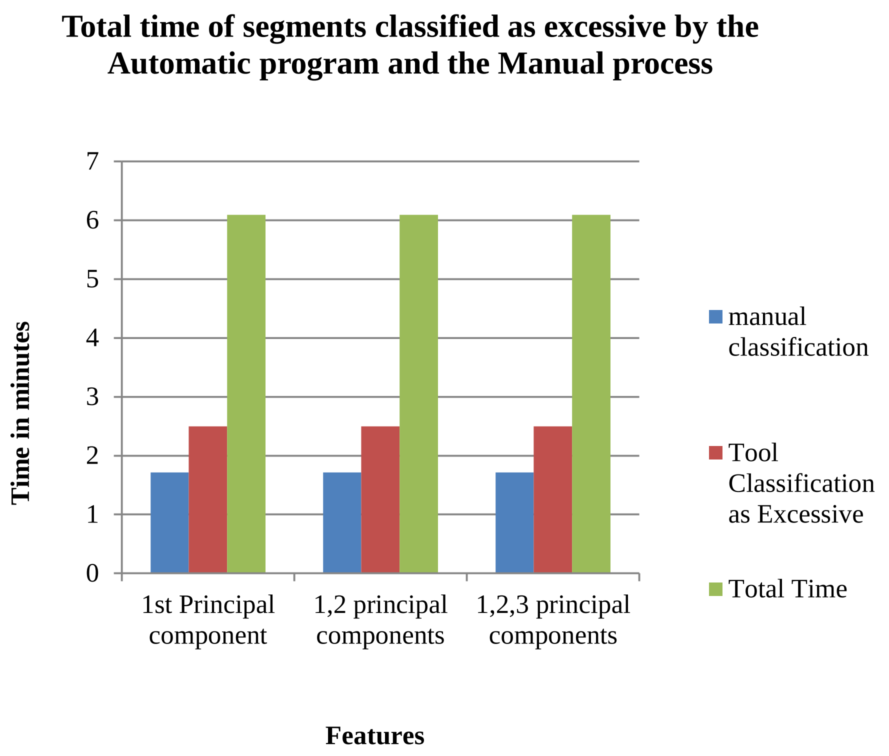
1st principal component

1st principal component, 2nd principal component

1st principal component, 2nd principal component, 3rd principal component

Data File 1





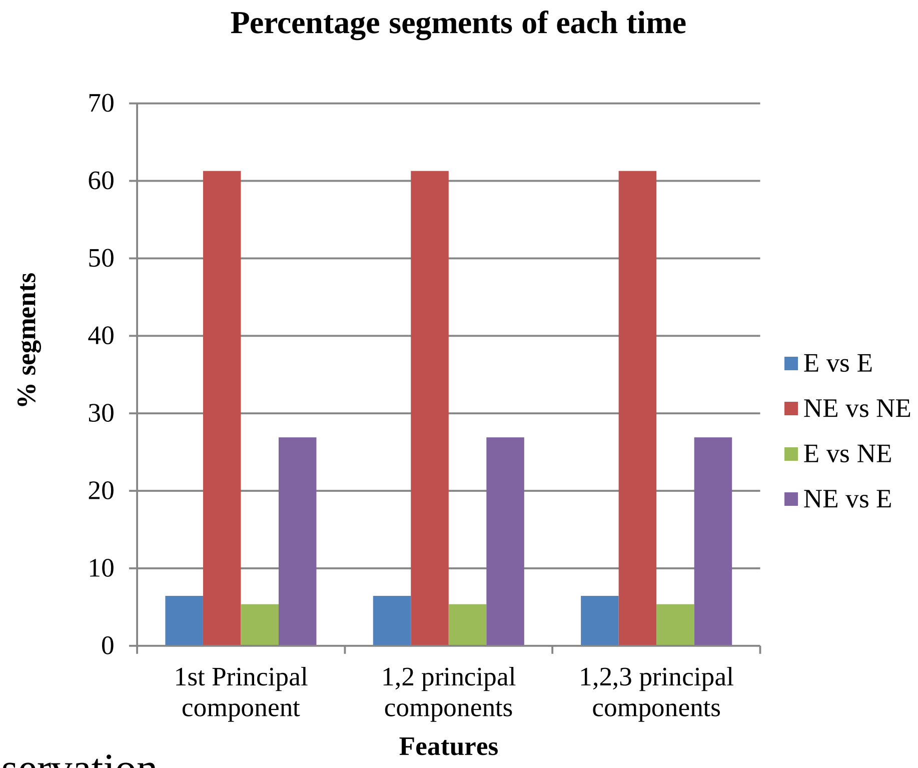
Observation

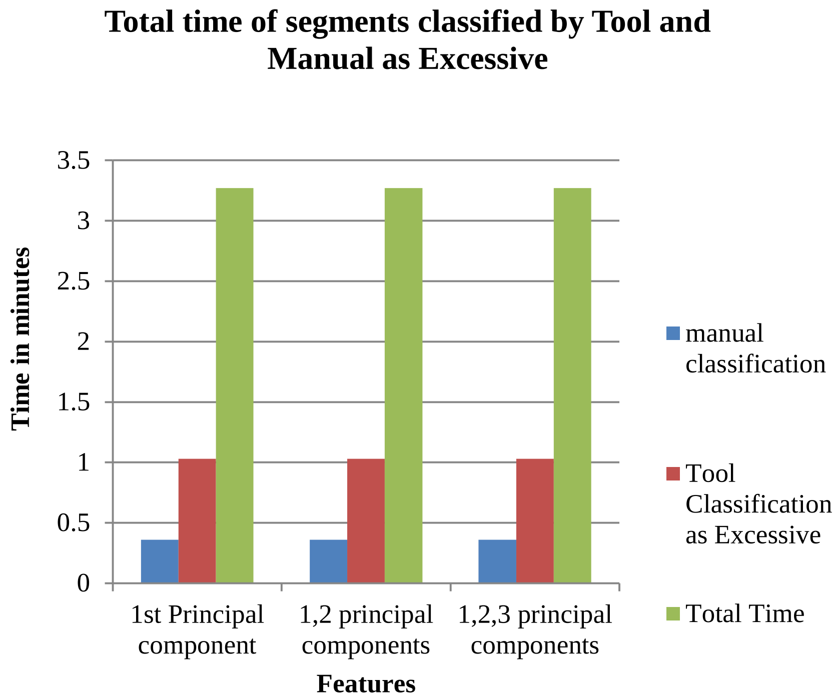
- All features have the same type-I and type-II errors

Observation

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

Data File 5





Observation

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

Observation

- 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

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

- 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

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

➢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

Feature value	avg. # of excessive effort segments	avg. total no of segments	avg. % type -I errors	avg. % type- II errors	avg. % of total errors	avg. Inspection time	avg. Inspection time as a % of total time
1st principal compone nts	16.6	95	27.5	4.1	31.6	2.7	61.2

- 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

Feature value	avg. # of excessive effort segments	avg. total no of segments	avg. % type- I errors	avg. % type- II errors	avg. % of total errors	avg. Inspection time	avg. Inspection time as a % of total time
1st, 2nd & 3rd principal compone nts	28.6	95	24.4	12.6	37.0	2.0	43.6

- 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

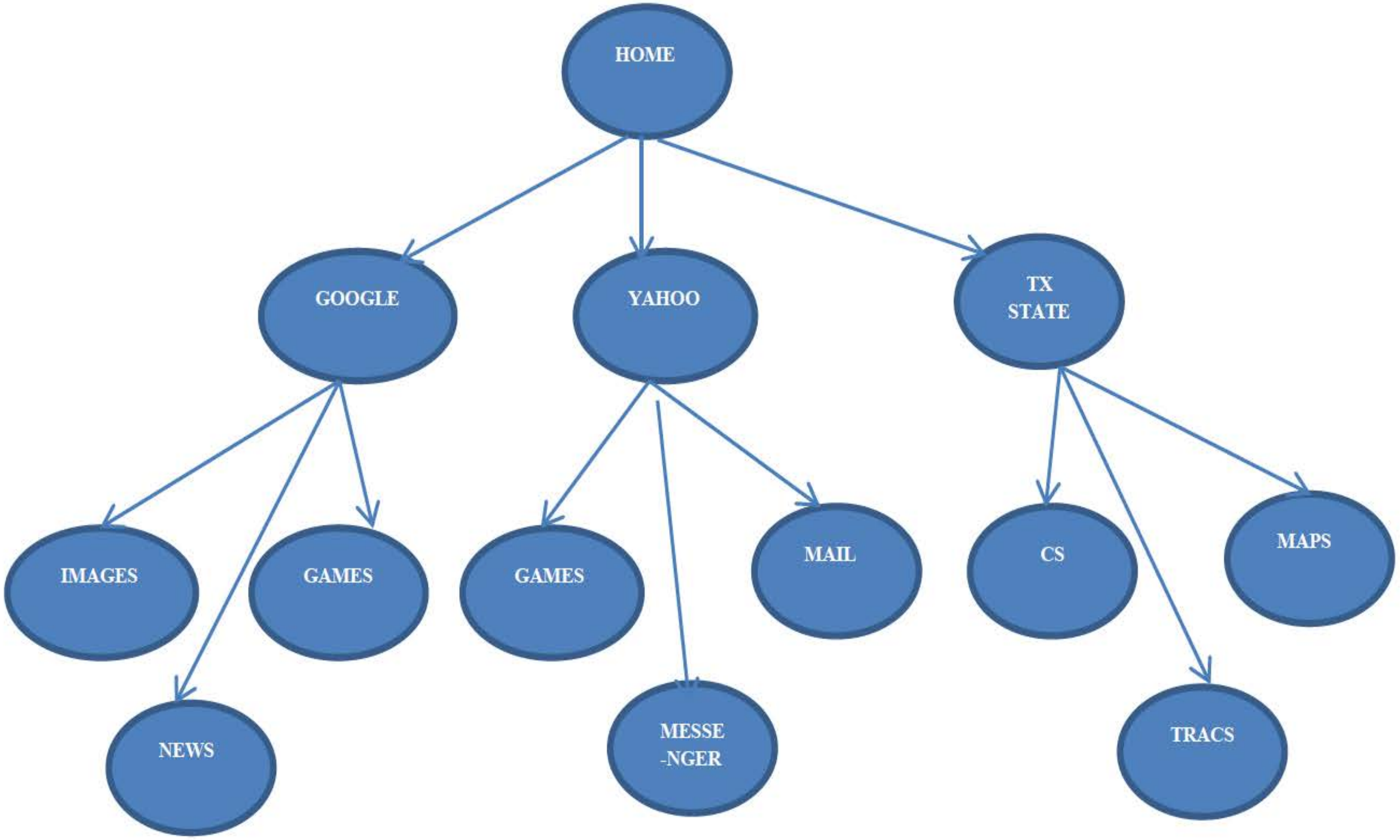
Physiological Emotion applications

- 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

r.

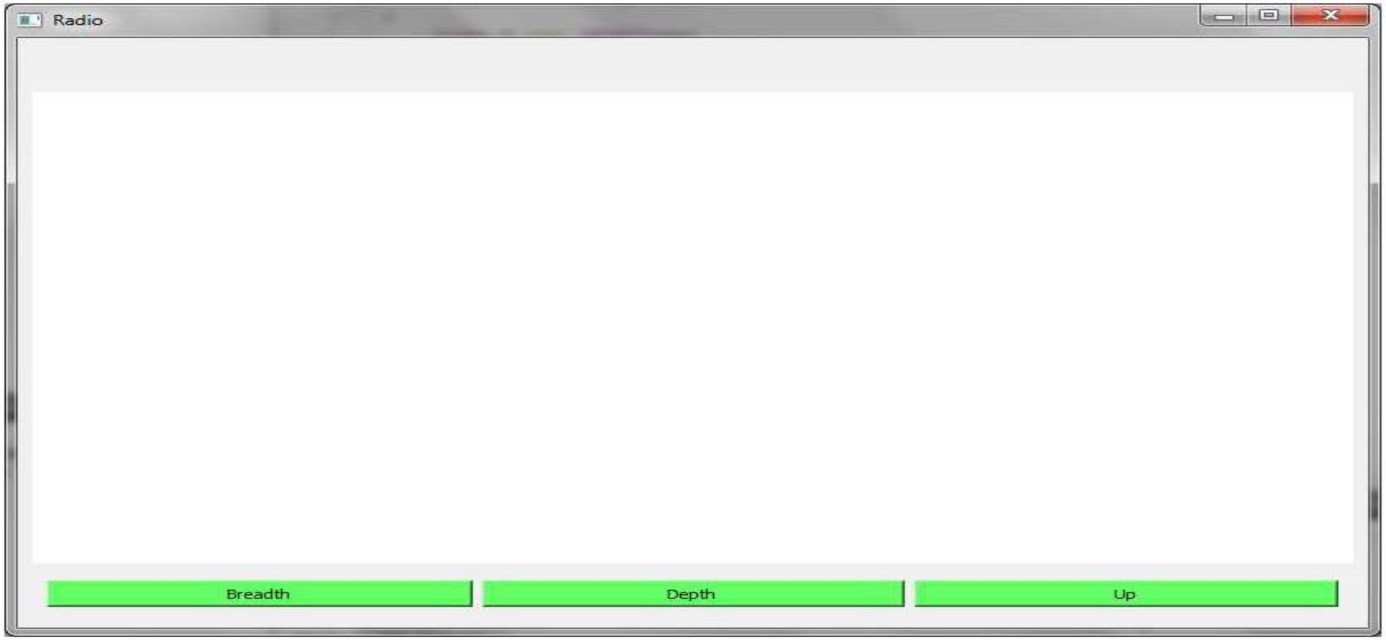
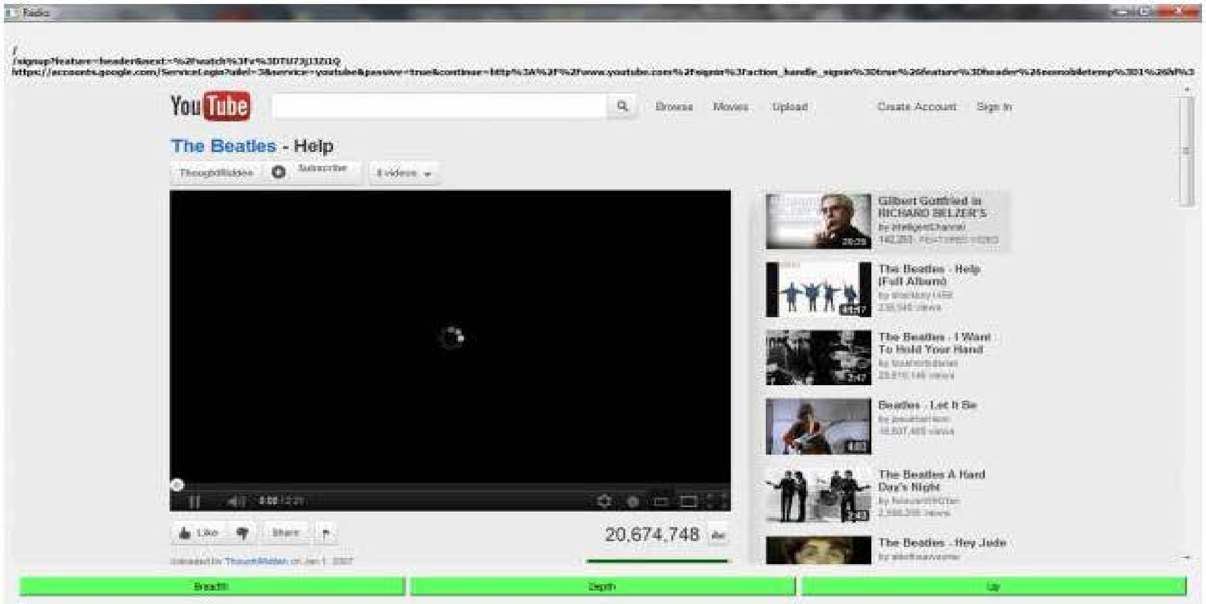


Figure 4: Initial screenshot of developed interface

Snapshots

The initial page is:



This page shows three URLs at the top left.