Design, Implementierung und Evaluation einer Frameworks für die Integration von Eye-Tracking-Signalen in Entwicklungsumgebungen.

Thema:
Design, Implementierung und Evaluation einer Frameworks für die Integration von Eye-Tracking-Signalen in Entwicklungsumgebungen.
Art:
MA
BetreuerIn:
Alexander Bazo
BearbeiterIn:
Jan Rankenhohn
ErstgutachterIn:
Christian Wolff
ZweitgutachterIn:
N.N.
Status:
in Bearbeitung
Stichworte:
Software Engineering, Eye-Tracking, Entwicklungsumgebungen
angelegt:
2016-07-11
Beginn:
2020-03-16
Textlizenz:
Unbekannt
Codelizenz:
Unbekannt

Hintergrund

Eye-Tracking entwickelt sich zunehmend zu einer neuen, intuitiven Methode der Mensch-Maschine-Interaktion. Die Technologie findet bereits im Bereich von Gaming, Automotive oder Medizin industrielle Anwendung. Eine besondere Rolle kommt ihr bei der Interaktion von Software zu. Entwickler können mit Hilfe der Blickdaten des Anwenders intuitive und natürliche Interaktionsmethoden erstellen und in ihre Anwendungen implementieren. Eine spezielle Art von Software sind Integrierte Entwicklungsumgebungen, die häufig eine Fülle von Funktionen bieten, um dem Entwickler möglichst viele Werkzeuge für die Programmierung an die Hand zu geben. Umso wichtiger ist hierbei ein gutes und durchdachtes Interaktionskonzept. Es existieren bereits einige Ansätze, um Eye-Tracking in IDEs zu integrieren. Neben der direkten Anwendungsinteraktion, können Blickdaten zudem Aufschluss über das Blickverhalten des Programmierers beim Lesen von Code

geben. Eine Integration von Eye-Tracking in IDEs kann über die Implementierung von Plugins oder Extensions, deren Entwicklung von praktisch allen Anbietern unterstützt wird, realisiert werden. Diese dienen als Schnittstelle zwischen Eye-Tracking Hardware und IDE System, indem sie die Blickdaten der Hardware aufnehmen und mit entsprechenden Funktionen der IDE verbinden.

Problemstellung

Problematisch bei der Entwicklung von Plugins ist die Diversität von IDEs auf der einen und Eye-Tracking Hardware auf der anderen Seite. Eye-Tracker unterscheiden sich von Hersteller zu Hersteller und auch modellabhängig in Bezug auf Softwareschnittstellen und Datenformat. IDE Plugins unterscheiden sich zudem bezüglich ihrer Programmiersprachen und Toolchains. Im ungünstigsten Fall ist ein entwickeltes Plugin in ausschließlich einer IDE und mit nur einem Eye-Tracker Modell verwendbar. Diese fehlende Flexibilität und Erweiterbarkeit steht einer intensiveren Nutzung und Entwicklung von Eye-Tracking Plugins in Industrie und Forschung im Wege.

Zielsetzung der Arbeit

Das Ziel dieser Arbeit ist es, ein Framework zu entwickeln, das als Schnittstelle zwischen Eye-Tracking Hardware und Plugin-Entwicklung fungiert. Aufgabe des Frameworks ist es, die Eye-Tracking Daten verschiedener Hersteller und Modelle aufzunehmen und einheitlich in Form eines dafür konzipierten Datenmodells zur Verfügung zu stellen. Die Anbindung an das Plugin soll plattformübergreifend möglich sein, um das Framework in den IDEs verschiedener Hersteller nutzbar zu machen. Zu Beginn der Arbeit soll eine Analyse der vorhandenen Eye-Tracking Modelle und deren APIs sowie der verfügbaren IDEs und Plugin Entwicklungsmöglichkeiten erfolgen. Zudem müssen mögliche Anwendungsfälle und bisherige Ansätze der Integration von Eye-Tracking in IDEs herausgearbeitet werden. Dazu ist auch eine genaue Betrachtung bisheriger Forschungsarbeiten in diesem Bereich nötig. Auf Basis dieser Analyse soll ein Konzept zur Entwicklung eines einheitlichen Datenmodells und zur Umsetzung des Frameworks erarbeitet werden. Die Verwendbarkeit und Funktionalität des Frameworks soll anschließend durch die Entwicklung eines Plugins evaluiert werden. Konkret soll das Plugin einen möglichen Use-Case implementieren (Heat Map - wie lange wurde etwas angeschaut, einfache Interaktion mit UI, ...) und mit mindestens 2 verfügbaren Eye-Tracker Modellen (z.B. TOBII) und 2 IDE Plattformen (IntelliJ, Visual Studio) getestet werden. Grundsätzlich soll das Framework Als zusätzliche Forschungsfrage kann untersucht werden, in wie weit günstige Eye-Tracker für die Verwendung im Rahmen der aufgezeigten Anwendungsfälle im Bereich der IDE Integration ausreichend sind.

Konkrete Aufgaben

- Übersicht über verbreitete IDEs und die mögliche Nutzung/Entwicklung von Plugins
- Recherche von Eye-Tracking-Standards und Plugin-Schnittstellen
- Recherche/Konzeption möglicher Anwendungsszenarien für den Einsatz von Eye-Tracking in Entwicklungsumgebungen
- Aufstellung allgemeiner Anforderungen für diese Szenarien hinsichtlich der Eye-Tracking-Hardware
- Übersicht vorhandener und nutzbarer Eingangs-Signale [x,y,Radius,Dauer,...] verschiedener

Eye-Tracker
- Implementierung eines Systems mit möglichst unabhängigen, erweiterbaren Komponenten
  - Hardware-Anbindung
  - Aufbereitung der Eingangs-Signale
  - Allgemeine Schnittstelle zum Bereitstellen der Daten
  - Plugin-Framework für eine IDE
- Entwicklung einer konkreten Demo-Anwendung auf der Basis des erstellten Systems

Erwartete Vorkenntnisse

- Sehr gute Kenntnisse des Software Engineerings
- Sehr gute Programmierkenntnisse
- Erfahrung mit C, C++ oder C#
- Gute Kenntnisse im Umgang mit IDEs
- Wünschenswert: Erfahrung in der Entwicklung von IDE-Plugins
- Wünschenswert: Hintergrundwissen zu Eye-Tracking

Weiterführende Quellen

- Rachel Turner, Michael Falcone, Bonita Sharif, and Alina Lazar. 2014. An eye-tracking study assessing the comprehension of c++ and Python source code. In Proceedings of the Symposium on Eye Tracking Research and Applications (ETRA '14). ACM, New York, NY, USA, 231-234. DOI=http://dx.doi.org/10.1145/2578153.2578218
- Hartmut Glücker, Felix Raab, Florian Echtler, and Christian Wolff. 2014. EyeDE: gaze-enhanced software development environments. In CHI '14 Extended Abstracts on Human Factors in Computing Systems (CHI EA '14). ACM, New York, NY, USA, 1555-1560. DOI=http://dx.doi.org/10.1145/2559206.2581217
- Bonita Sharif and Huzefa Kagdi. 2011. On the use of eye tracking in software traceability. In Proceedings of the 6th International Workshop on Traceability in Emerging Forms of Software Engineering (TEFSE '11). ACM, New York, NY, USA, 67-70. DOI=http://dx.doi.org/10.1145/1987856.1987872
- S. Yusuf, H. Kagdi and J. I. Maletic, „Assessing the Comprehension of UML Class Diagrams via Eye Tracking,“ 15th IEEE International Conference on Program Comprehension (ICPC '07), Banff, Alberta, BC, 2007, pp. 113-122. DOI=http://dx.doi.org/10.1109/ICPC.2007.10