Projektplan 1.0 – Vision Control

Idee

Die Regelung mechatronischer Systeme erfordert eine sehr genaue Messung der zu regelnden Größen. Das Designen, Implementieren und Kalibrieren dieser Sensoren nimmt dabei nicht nur einen sehr großen Zeitanteil in Anspruch, sondern ist auch im großen Maße für die Qualität der Regelung zuständig.

Ein besonders interessanter Ansatz zur alternativen Lösung des Sensorproblems ist die Verwendung selbstlernender Systeme. Die Idee dahinter besteht in ihrem Kern aus einer Verschiebung der Komplexität aus der Modellierung (Knowhow des Ingenieurs) in die Daten selbst.

Eine Performancesteigerung liefert diese Methode vor allem bei sehr verrauschten Daten, oder solchen, bei denen gleichbleibende Messbedingungen nicht gegeben sind. Dies wird ermöglicht durch die Fähigkeit dieser AI Systeme die Trainingsdaten nicht nur „auswendig“ zu lernen, sondern darüber hinaus auch zu verallgemeinern. Dies ermöglicht einen Wissens Transfer aus den Trainingsdaten in die Echtzeitanwendung auf komplett neue Daten.

Wir wollen dabei vor allem zeigen, wie durch den Einsatz einer Kamera, eines leistungsstarken Rechners und Trainingsdaten aus einem wohlüberlegten Versuchsaufbau, viele Sensorlösungen in der Regelungstechnik ersetzt werden können.

Wir erhoffen uns dabei zum einen hinreichend genaue Messdaten in hinreichend kurzer Zeit zu bekommen und zum anderen vielleicht sogar eine Messwerterfassung eines mechatronischen Systems, das mit erprobten Methoden nur durch sehr hohen Aufwand zu messen ist,

Hardware

- Einen leistungsstarken Rechner für das Training der Netzwerke
- Einen leistungsstarken Embedded PC für die Inferenz in Echtzeit
- Ein mechatronisches System als Ausgangspunkt (Quanser Hardware)
- Eine schnelle und einfache USB Kamera für einen ersten Prototypen
- Eine Hochauflösende Tiefenkamera für ein hoch performantes Endprodukt

---

Stage-Gate Plan

Stage 1: Projektplanung, Hardwareentscheidungen und Recherche

In dieser Phase geht es vor allem darum Fehlentscheidungen, die zu einer schlechten Planung und Zielsetzung führen, zu vermeiden. Außerdem soll verhindert werden, dass durch zu frühem Konzentrieren auf spezielle Hardware schlechtes Zeitmanagement betrieben wird.

Da das Projekt von Anfang an als Lernprojekt angesetzt ist, legen wir großen Wert darauf frühzeitig vorzeigbare Ergebnisse vorzuweisen. Ein weiteres Problem stellt dabei auch unsere fehlende Erfahrung mit den geplanten Methoden dar. Dies erschwert vor allem die Zeiteinschätzung. Wir lassen uns daher vor allem für die Endphase des Projekts viel Raum für Anpassungen und Weiterentwicklungen.

Entscheidend ist in dieser Phase auch, dass wir uns schnellstmöglich mit dem Deep Learning Framework Tensorflow bekanntmachen und sicher erste Programme schreiben können.

Stage 2: Machbarkeitsnachweis - Uhrenstand ablesen durch CNN

Ein schöner Schritt bevor wir einiges an Zeit und Ressourcen in die Datensammlung stecken ist ein erster Machbarkeitsnachweis mit einem passenden Datensatz aus dem Internet eine ähnliche Aufgabe zu lösen. Beispielsweise das ablesen eines Zeigerstandes einer Analoguhr durch ein CNN.

Dabei spielt vor allem der Wissensgewinn and die Hardwareanforderungen eine entscheidende Rolle. Erkenntnisse, die hier gewonnen werden sind entscheidend für die Neuplanung and Adaptierung der folgenden Stages.

Gate 2

Die Messwerterfassung aus dem vertretbar großen Trainingsset funktioniert ausreichend genau. Wichtig ist hier auch, dass weder die Trainingszeit, noch die Inferenz zu lange dauert, was den Fortschritt in den folgenden Stages gefährden könnte.

Stage 3: Kamera ansteuern und abstrahieren

In diesem Schritt geht es um das Bekanntmachen mit openCV und der Kamerahardware. Ziel ist dabei nicht nur die Videoaufnahme, sondern vor allem den Zugriff auf die einzelnen Pixelwerte in Echtzeit aus einem Python Skript. Ein schönes Ergebnis wäre dabei eine Abstrahierung der Kamera, die einen leichten Wechsel auf eine andere Kamera ermöglicht.

Stage 4: Versuchsaufbau und Datensammlung

Waren Stage 2 und 3 erfolgreich geht es nun um eine genaue Versuchsplanung. Ziel ist es, brauchbare Trainingsdaten zu sammeln, um ein erstes brauchbares Netzwerk zu trainieren.

Dabei ist wichtig, dass wir genug Daten sammeln, die es einem Netzwerk erlauben ausreichend zu verallgemeinern.

Gate 4

Die Daten wurden so aufbearbeitet, dass sie leicht aus einem anderen Skript geladen werden können.

Stage 5: Basic Model erstellen

Mit den Daten aus Stage 4 wird nun ein erstes einfaches Netzwerk trainiert und untersucht. Dabei liegt der Fokus erstmal nicht auf Performance, sondern auf den Nachweis, dass diese Methode vielversprechend ist und sich das Hyperparameter Tuning überhaupt lohnt.

Gelingt dieser Nachweis nicht, muss über eine andere Modellarchitektur nachgedacht werden.

Stage 6: Hyperparameter Tuning

Die Hyperparameter des Modells aus Stage 5 werden so lange optimiert, bis es eine ausreichende Genauigkeit aufweist. Falls es die Komplexität des Modells erlaubt ist dabei auch zu beachten, dass diese Genauigkeit durch ein möglichst kleines Modell erreicht wird, damit die Inferenzzeiten nicht zu groß werden.

Gate 6

Die Ergebnisse, die bis zu diesem Punkt erreicht wurden sind maßgebend für das weitere Vorgehen des Projekts, dabei sind mehrere Szenarien möglich:

- Das Modell ist zwar sehr genau, aber alles deutet darauf hin, dass die großen Inferenzzeiten keine Implementierung auf Embedded Hardware zulässt. Trifft dies zu streben wir eine Konzentration der Teamressourcen auf die eigene Platine.
- Das Modell ist zwar schnell, hat aber noch einen feststellbaren systematischen Fehler, der wahrscheinlich durch eine neue Modellentwicklung mit einer hochauflösenden Tiefenkamera beseitigt werden kann. Hier würde sich der Aufwand der Einarbeitung in die neue Hardware und neuer Trainingsdatensammlung lohnen.
- Das Modell ist sowohl genau, als auch schnell. Hier ist zu überlegen, ob nicht schon mit bestehender Kamerahardware auf Embedded Hardware umgestiegen wird.

Stage 7: Umstieg auf ZED

Ziel ist eine neue Iteration der Modellentwicklung mit wesentlich mächtigerer Hardware. Dabei gilt zu klären, ob man eventuelle Transfer-Learning Techniken einsetzten kann um den Bedarf an neuen Trainingsdaten zu minimieren.

Ziel ist hierbei vor allem eine Steigerung der Genauigkeit. Wichtig ist dabei, dass die Inferenzzeiten handhabbar bleiben.

Stage 8: Go Embedded

Ziel dieser Phase ist ein erfolgreiches implementieren des Modells auf einen Embedded Computer. Dabei stehen vor allem die Schnelligkeit und die Kompaktheit des Systems im Vordergrund.

Ist dieser Schritt einmal gelungen, lässt sich von diesem Embedded PC auch die Regelung übernehmen.

Stage 9: Platinendesign