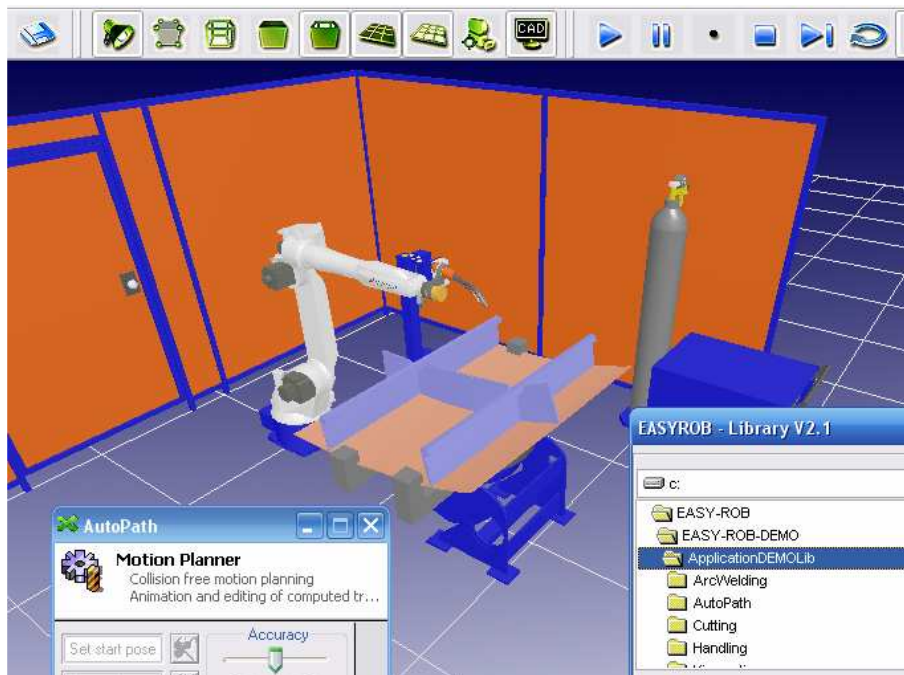


Update

EASY-ROB™ V4.307



August 2007

Version 1.01

EASY-ROB™

Inhaltsverzeichnis

Einleitung	3
Allgemeines.....	4
Dual-Core - Fähigkeit	4
Windows VISTA™ - Fähigkeit	4
Kinematiken mit weniger als 6 unabhängigen Achsen.....	4
API Funktionen	4
Neue Bibliotheken.....	5
ApplicationLib	5
DeviceLib.....	5
TrainLib.....	5
ERPL-/ ERCL-Beispiele	5
Neue Funktionen / ERCL-Kommandos.....	6
If – While - Goto.....	6
Neue Option	8
AutoPath™ - Kollisionsfreie Bahnplanung	8
Eigene Notizen.....	9

EASY-ROB™ V4.307

Einleitung

Die aktuelle Version EASY-ROB™ V4.307 ist das Ergebnis der letzten Monate dieses Jahres und auch die letzte Zwischenversion aus V4.3.

Viel Entwicklungszeit wurde investiert um EASY-ROB™ Dual-Core fähig zu machen. Auch erste Tests unter Windows Vista™ sind positiv gelaufen. Somit blieb für die Weiterentwicklung neuer Funktionen weniger Zeit als ursprünglich geplant.

Trotz allem wurde der ERPL-/ERCL- Befehlsumfang erweitert, es sind neue API-Funktionen hinzugekommen und das numerische Lösungsverfahren wurde verbessert. Darüber hinaus wurden auch einige Fehler beseitigt und Kleinigkeiten verbessert.

Stolz sind wir auf die neue Option AutoPath™, die im Rahmen eines Kooperationsprojektes mit der Hochschule Darmstadt entwickelt wurde. Immer dann wenn es besonders eng ist, können mit AutoPath™ kollisionsfreie Bahnen erzeugt werden. Die Berechnung mittels RRTs (Rapidly Random Trees) kann zwar einige Minuten dauern, erleichtert aber die Arbeit für den Bediener erheblich. Teilweise sind die Lösungen derart überraschend, dass man als „Mensch“ nicht so ohne weiteres darauf gekommen wäre. Ein optimales Ergebnis wird erzielt, wenn der Bediener sein Prozess Know How mit AutoPath™ kombiniert.

Diese Version ist wie immer nur ein weiterer Meilenstein. Die Liste der zu entwickelnden Funktionen ist lang, was auch damit zusammenhängt, dass EASY-ROB™ als DLL Version sehr kundenspezifisch eingesetzt wird und unsere Unterstützung erfordert. Auch der EASY-ROB™ Simulations Kernel, der vom Offline Programmiersystem Famos robotic® verwendet wird, wird permanent weiterentwickelt. Vom Kernel gibt es auch seit kurzem eine Version unter Linux.

Unsere Entwicklung geht weiter – immer das Ziel vor Augen ein Tool zu liefern, welches das Planen und Simulieren vereinfacht und den Nutzer in jeder Hinsicht unterstützt.

Die zukünftige Version V4.6 soll es dann endlich ermöglichen mehrere Roboter gleichzeitig zu simulieren. Jeder Roboter hat dann ein eigenes Programm. Roboter können dann mittels I/O-Signalen kommunizieren und synchronisiert werden. Diese Funktionalität erfordert größere Änderungen der internen Softwarestruktur, die zum Teil schon implementiert ist.

An dieser Stelle möchten wir uns bei unseren Kunden und Anwendern bedanken, die mit ihren Vorschlägen und Anforderungen zur Entwicklung beigetragen haben.

Vielen Dank

A handwritten signature in blue ink, appearing to read 'Stefan Anton'.

Stefan Anton

EASY-ROB
3D Robot Simulation Tool

Allgemeines

Dual-Core - Fähigkeit

Die fortschreitende Entwicklung im Hardwarebereich geht ungebremst weiter, um den permanent steigenden Anforderungen gerecht zu werden. So erfordern Änderungen in der Hardwarearchitektur auch wieder Anpassungen innerhalb der Software, um unerwünschte Effekte (die plötzlich und unregelmäßig auftreten können) auszuschalten.

Einer dieser Effekte ist das „Umherspringen von Geometrien“ beim grafischen Update in EASY-ROB™. Dieses Phänomen trat nur bei Rechnern mit Dual-Core- und Multi-Core-Prozessoren auf. So sind die Threads bei der Programmsimulation nicht einwandfrei synchronisiert abgelaufen.

Dies war auch ein Thema bei „EASY-ROB™ auf Dual-Core“.

Aber die Investition von vielen Stunden der Entwicklungsabteilung hat sich an zwei Fronten ausgezahlt:

1. „ruckelige“ Simulationen, bei denen auf Dual-Core Rechnern beim grafischen Update mitunter ganze Maschinengeometrien „durch die Arbeitszelle springen“ gehören der Vergangenheit an und
2. nebenbei verzeichnen wir eine signifikante Verbesserung der grafischen Performance

Windows VISTA™ - Fähigkeit

Nicht nur die Hardware unterliegt derzeit großen Änderungen, sondern auch das Betriebssystem. Die Frage, inwieweit VISTA™ die aktuelle OpenGL Version unterstützt ist weiterhin nicht vollständig geklärt. Erste EASY-ROB™ Tests unter Windows Vista™ waren durchweg positiv. Allerdings haben wir bisher keine Performance-Messungen durchgeführt.

Kinematiken mit weniger als 6 unabhängigen Achsen

Zur Lösung des inversen kinematischen Problems für Roboter mit weniger als 6 unabhängigen Achsen (global degenerierte Kinematiken) hat der EASY-ROB™ Bediener die Möglichkeit eine eigene Lösung in der DLL „er_kin.dll“ in der Programmiersprache C zu schreiben. Für einen ersten „Schuss“ reicht allerdings das numerische Lösungsverfahren „Numerical Inv.Kin.+ Data“ oft aus, wobei damit auf alle möglichen Konfigurationen verzichtet wird. Für global degenerierte Kinematiken wurde das Verfahren verbessert, in dem die bevorzugte Tool-Achse (meistens die Z „Approach“-Achse des TCPs) erreicht wird. Hierbei ist beispielsweise die Drehung um diese Z-Achse nicht bestimmt. Bei diesen Kinematiken sollte die Sub-ID = 1 gewählt werden.

API Funktionen

Jede Menge neue API Funktionen sind hinzugekommen. Schauen Sie bitte in den Header-Dateien „./er_dvlp/er_dvlp.h“ und „./er_dvlp/er_dvlp_ext.h“ nach.

Neue Bibliotheken

Die neuen Bibliotheken stehen ab sofort auf der neuen [EASY-ROB™-CD](#) und im Kundenbereich der EASY-ROB™ Webseite zur Verfügung.

ApplicationLib

Die Beispiele der ApplicationLib zeigen Anwendungen aus den verschiedensten Bereichen. Vom Handling über das Bahnschweißen bis zum Lackieren. Allesamt Anwendungen, die mit EASY-ROB™ geplant und untersucht werden können.

DeviceLib

Die DeviceLib beinhaltet diverse Geräte wie Förderbänder, Greifer, Schweißpistolen, Tische, Stühle, etc. Alle Devices sind als ROB-Files angelegt und sind somit z.B. automatisch in den Kollisionsketten berücksichtigt.

TrainLib

Die TrainLib und das dazugehörige Tutorial unterstützen den Benutzer bei den ersten Schritten in EASY-ROB™. Wie entlang eines roten Fadens geht's von der leeren Arbeitszelle zur fertigen Simulation.

Die Dateien befinden sich im Kundenbereich auf der Webseite von EASY-ROB:
<http://www.easy-rob.com/download/easy-rob/manual-beispiele>

ERPL-/ ERCL-Beispiele

ERCL (EASY-ROB Command Language) ist eine Erweiterung der ERPL (EASY-ROB Programming Language) und automatisiert nahezu alle Interaktionen des Users in einem Roboterprogramm. Ob nun das Ein-/Ausschalten der TCP-Spur, das Ein-/Ausblenden von Objekten oder das Verändern der Simulationsgeschwindigkeit – um nur einige Beispiele zu nennen – mit ERCL erzeugen Sie fortschrittliche und effektive Simulationen.

Die Datei „Proj_Example_ERPL.zip“ enthält viele Beispiel-Arbeitszellen anhand derer die Funktionen der ERCL/ERPL erläutert werden. Rufen Sie einfach nur die entsprechende Arbeitszelle auf und starten Sie die Simulation. Im Programmfenster können die verwendeten Kommandos nachvollzogen werden.

Die Datei „Proj_Example_ERPL.zip“ mit zugehöriger Dokumentation befindet sich Download Bereich:
<http://www.easy-rob.com/download/easy-rob/manual-beispiele>

Neue Funktionen / ERCL-Kommandos

If – While - Goto

Mit den neuen IF, WHILE und GOTO –Statements ergeben sich neue Möglichkeiten innerhalb des Roboterprogramms. Die Statements IF und WHILE prüfen eine Bedingung auf „True“ und „False“. Dabei ist die Bedingung ein mathematischer Ausdruck und ist wahr, wenn größer Null.

So kann zum Beispiel ein Roboter mit einer „**IF-Abfrage**“ im Ablauf entscheiden zu welchem Förderband er fährt, um Ware aufzunehmen bzw. diese abzulegen.

Beispiel IF-Statement: $xx > yy$, somit wird der Roboter zu Tag 'T_1' fahren

```
xx=3; yy=1
IF gt(xx,yy)
    LIN T_1
ELSEIF lt(xx,yy)
    LIN T_2
ELSEIF eq(xx,yy)
    LIN T_3
ELSE
    ! ERPL,ERCL hier einfügen
ENDIF
```

Bei Nutzung einer **WHILE-Schleife** fährt der Roboter zum Beispiel solange eine Strecke ab, bis eine Bedingung nicht mehr wahr ist.

Beispiel WHILE-Statement: Der Roboter fährt den Pfad 'path01' dreimal entlang.

```
xx=4; yy=1
WHILE gt(xx,yy)
    xx = xx-1
    ALONG PATH01 T_1 T_5
ENDWHILE
```

Siehe Beispiele in „./Tutorial/Proj_example_erpl/“
functions_if_while_goto_01.cel
functions_if_while_goto_02.cel
functions_if_while_goto_03.cel
functions_if_while_goto_04.cel

Und mit einem **GOTO** kann - wie in allen anderen Programmiersprachen – zu einer anderen Stelle im Programm gesprungen werden. Es obliegt hier dem Bediener, GOTO „sinnvoll“ zu verwenden.

Beispiel:

```
GOTO LABEL my_label
```

...

! dieser Bereich wird übersprungen

...

```
LABEL my_label
```

Eine ausführliche Dokumentation der Befehle und Beispiele befindet sich im Kundenbereich:

<http://www.easy-rob.com/download/easy-rob/manual-beispiele>

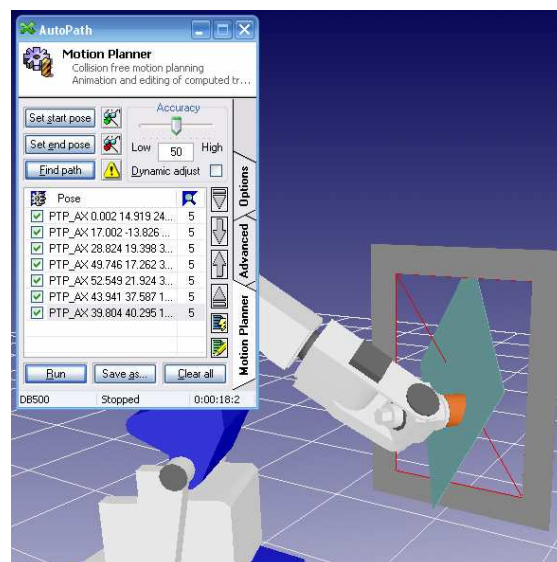
Neue Option

AutoPath™ - Kollisionsfreie Bahnplanung

Das Erzeugen von Bewegungspfaden zum kollisionsfreien „Einfädeln“ von Werkstücken in die Vorrichtung oder zum Umfahren von Störkonturen ist in der Regel ein sehr zeitaufwendiges Unterfangen.

Die neue EASY-ROB™ Option **AutoPath™**, mit deren Hilfe der Bediener kollisionsfreie Bahnen bzw. Pfade erzeugen kann, unterstützt bei den oben genannten Aufgaben.

Die Beispielarbeitszellen zum Thema kollisionsfreie Bahnplanung befinden sich in dem Unterverzeichnis „AutoPath“ in der „ApplicationLib“ bzw. der „ApplicationDEMOLib“.



Wie funktioniert AutoPath™?

Im ersten Schritt wählt der Bediener eine kollisionsfreie Start- und Zielposition, bei der auch die Verfahrbereiche nicht überschritten sind. Im zweiten Schritt wird mit „Find path“ eine kollisionsfreie Bahn ermittelt. Hierbei werden mehrere achsspezifische Stützpunkte generiert und im AutoPath™-Dialog angezeigt. Diese können beliebig in das Roboterprogramm im Teach-Window eingefügt werden.

AutoPath™ wurde im Rahmen eines Kooperationsprojektes mit der Hochschule Darmstadt, Fachbereich Informatik (Prof. Dr. T. Horsch, Rudi Scheitler) entwickelt.

Die Beschreibung zu dem Beispiel befindet sich im Downloadbereich:

<http://www.easy-rob.com/download/easy-rob/manual-beispiele>



Update EASY-ROB™ V4.307

Eigene Notizen