Vorgehensweise und Programmanleitung

Wichtiges zum Export von .rad Dateien in Hypermesh v.9.0:

- **.iges –File** (Konstruktion) des Fahrzeugs in Hypermesh importieren. (Import Type = Geometry, File Type = RADIOSS(Block), .iges – Filepfad angeben
- Prüffeldlinie/n markieren und 1D – Line Mesh durchführen, **Abstand: 10mm**
- Exportieren der Linienpunkte als **.rad File**, Export Type: FE-Model, FileType: Radioss, Block 51, File benennen

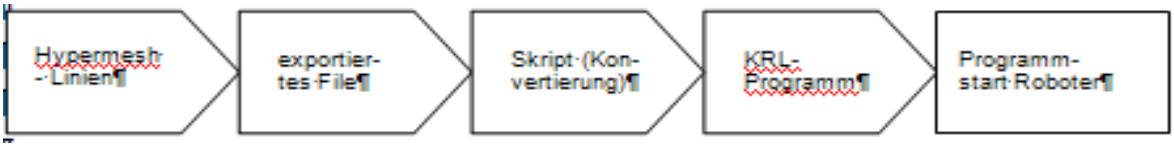
Dieses Programm bzw. Tool „RAD2KRL“ ist kompatibel mit Altair HYPERMESH v9.0 & KUKA Roboter Modell KR140 L100.

Beschreibung Tool/Programm „RAD2KRL“

Damit das Programm überhaupt läuft, ist vorher „Python 3“ (Version 3.0.1) zu installieren!!! Download und anschließendes installieren auf
http://www.python.org/download/releases/3.0.1/ oder von CD!
Für Windows x86 MSI Installer auswählen.

Dieses Tool erzeugt KRL(KUKA Roboter Language) Code bzw. Bewegungsanweisungen, damit Prüffeldlinien von Prüffeldern wie Phase I, II und ENCAP automatisiert auf eine beliebige Fahrzeugfront übertragen werden können.
Es dient als Schnittstelle zwischen dem Programm Hypermesh von Altair am Rechner und der Robotersteuerung (KRC2), die die zu zeichnenden Prüffeldlinien als Bewegungsanweisungen für den verwendeten KUKA-Roboter interpretieren muss.

- Aus Hypermesh werden ausgehend von dem Konstruktionsmodell des Fahrzeugs die Linienpunkte des Prüffelds als Punkte in der Form P: (X,Y,Z) in einem Abstand von (**OPTIMAL: 10 MM**) in ein Plaintext-File (Endung '.rad') exportiert.
- Dieses Tool konvertiert jedes exportierte .rad-File in einen für die Robotersteuerung verständlichen KRL-Code, d.h. in ein Programmfile (mit Endung '.src'), das nach erfolgter Konvertierung in das Filesystem der Robotersteuerung kopiert wird.
- Damit lässt sich nun am Roboter das kopierte KRL-Programfile starten und der Roboter führt Bewegungsanweisungen in der Reihenfolge der exportierten Punkteliste (=.rad File) durch. Mit der am Roboterarm/Effektor angebrachten Lackstifthalterung lassen sich die Linien einzeichnen. Voraussetzung dafür ist, dass man sich mit dem Roboter korrekt in das Base-(Fahrzeug)und das Tool-(Lackstift) Koordinatensystem eingemessen hat.
- Schnittstellen: Rechner(Hypermesh) <= Rechner(Tool) => Roboter(KUKA/KRL)bzw. Robotersteuerung(KRC2) - Fahrzeug
- Konvertierungsprozess: Hypermesh (.rad File) => Tool => KRL Programmfile (.src File) - Kopie des/der .src File(s) in das Filesystem der Robotersteuerung mittels USB, Ethernet, Diskette.



Wichtiges zum Ablauf des Programms und zur Programmbenutzung:

- Die Python3-Umgebung installieren (Ordner „pyinstaller“). Zum Ausführen des Programms die .py Datei im Ordner „pyprogram“ der CD doppelklicken. Der CD sind im Ordner „kukaconfig“ Konfigurationsdateien beigelegt. Diese nur einspielen bei Roboter-Reset oder einem fabrikneuen Roboter. Der CD sind im Ordner „kukaconfig“ Konfigurationsdateien beigelegt. Diese nur einspielen bei Roboter-Reset oder einem fabrikneuen Roboter.
- Das Programm bietet die Möglichkeit, einen Ordner auszuwählen, wo die zu konvertierenden .rad Files liegen. Diese .rad Files werden eingelesen und in .src Files konvertiert (Dateiname des .src-Files: KRL/KUKA-seitig beschränkt auf **max. 24 Zeichen**, ohne Sonderzeichen!). Der Dateiname des .rad-Files wird für das daraus konvertierte .src-File übernommen. Jedoch ist ein Dateiname eines extrahierten .rad-Files länger als 24 Zeichen, wird der Dateiname des konvertierten .src-Files automatisch auf max. 24 Zeichen gekürzt. Wird ein Ordner ausgesucht, wo sich keine .rad Files befinden, wird man aufgerufen, einen neuen Ordner auszuwählen. Hier bricht bei „Cancel“ das Programm ab.
- Das Format für neue, konvertierte .src Files ist 'KRL_...' alter Filename von .rad. Die neuen .src Files werden automatisch in einem Unterordner in einem ebenfalls selbst gewählten Verzeichnis (oder standardmäßig in das Home-Verzeichnis des Users bei Abbruch des zweiten Aufrufs zur Ordnerauswahl) kopiert. Dieser Ordner bekommt automatisch den Namen im **Format 'srcFolder_ddmmyy_hhmmss'** zugewiesen, um die Eindeutigkeit des Ordners sicherzustellen (d=Tag, m=Monat, y=Jahr, h=Std, m=Min, s=Sec). In diesem Ordner liegen nun die neuen, generierten .src Files für die Robotersteuerung des KUKA-Roboters. Hat man keine Schreibrechte auf das gewählte Verzeichnis, wird man erneut aufgerufen.
- **Den Dateinamen dieser neu generierten .src Files nicht mehr umbennen!!! Der Dateiname des .rad Files hingegen kann vor der Konvertierung umbenannt werden.**
- Diese .src/bzw. Programmfiles sind nun in das Filesystem des Roboters/der Robotersteuerung zu kopieren und das Programmfile kann gestartet werden. Der Roboter sollte nun die Bewegungsanweisungen nach den exportierten Punkten des/der .rad File/s durchführen.
- Die Rotationen um die XYZ-Achsen, so dass der Lackstifts immer flächennormal zeichnet, werden automatisch berechnet. (z.B. bei Krümmungen der Fronthaube). Werden nur einzelne Punkte auf der Fronthaube angefahren oder tritt ein Fehler auf, wird erfolgt keine Rotation des Stifts.
- Das Programm kann natürlich nur genau die Punkte abfahren, die auch in das .rad File von Hypermesh exportiert worden sind, z.B. ein ganzes Prüffeld oder auch nur einzelne Linien...

WICHTIG:

- Das Programm ist darauf ausgelegt, eine Punkteliste von 10mm (=optimal) Abständen zu verarbeiten. Nur dann werden Linien optimal und möglichst genau (Abweichung CAD/CAE Hypermesh/Bauteiltoleranzen etc.) eingezeichnet.
- **Ab Abständen von 20mm werden Punkte nur mehr einzeln abgefahren**, ohne Linieneinzeichnung, das Programm schaltet in den **Punktemodus**. Wenn gewünscht, kann eine Linie auch strichliert eingezeichnet werden (nötig bei ENCAP-Prüffeld). Dazu ist es nötig, die (strichlierten) Linien eines Prüffelds **einzeln** zu extrahieren und beim Einlesen des .rad-Files. in dem sich die

Koordinaten der strichlierten Linie befinden, **'Linie strichliert zeichnen?' mit 'JA' zu bestätigen.** Phase I/II Prueffelder ganz einfach normal als gesamtes Prueffeld extrahieren.

- Sinnvoll ist es, den Dateinamen des extrahierten .rad Files mit z.B. 'strichliert' zu kennzeichnen, um die Punkteliste der Linie bzw. das .rad File selbst leichter zu erkennen, das strichliert werden soll. Die Lücken auf der strichlierten Linie entsprechen einem doppelten Punkteabstand.

Folgende Optionen sind also möglich:

1) Linien normal einzeichnen (Export der Linienpunkte in Hypermesh im Bereich von [min. 1mm bis max: 20 mm] – **optimal sind 10 mm**!
2) Linien strichliert einzeichnen (nur bei ENCAP- Prueffeld - .rad Dateiname umbenennen/kennzeichnen mit z.B. 'strichliert' und im Programmablauf bei dieser .rad Datei mit 'JA' bestätigen)
3) Punkte abfahren - es werden keine Linien eingezeichnet, sondern nur einzeln Punkte abgefahren - dazu ist es nötig, die Punkte im Abstand von grösser als >20mm zu exportieren. Bei 2) innerhalb der strichlierten Linie und bei 3) zwischen den einzelnen Punkten verfährt der Roboter immer wieder auf die HOME-Position und wieder zurück, für Anfangskalibrierung und bei Ende auch - das bei 1), 2) und 3).

Ablauf am KUKA Roboter (Modell KR140 L100)

- Diskette/USB Stick anschließen
  - Für USB ggf. KUKA-Softwareversion updaten!
- Kopieren/Einspielen der „.src“ – Programmdateien (KRL – KUKA Roboter Language) vom USB Stick/Diskette (Laufwerk A:) ausschließlich nur **in den Ordner KUKA/R1/Program – Wichtig, damit das Programm läuft!**
- Einmessen/Auswahl in korrektes Fahrzeugkoordinatensystem, wo Prüffeldlinien eingezeichnet werden sollen (Nummer merken)
- Einmessen/Auswahl von Werkzeugkoordinatensystem (Lackstift/Halterung) – Nummer merken
- Programmfile von Linie **KRL_...** anwählen
- Totmannschalter (weiß, hinten) und **grüne Taste (+) (oder grüner Knopf** hinten) gleichzeitig drücken, damit das Programm durchgefahren wird
- Programmcheck: Damit das Programm weiter läuft noch gewünschtes Werkzeug-KS und Fahrzeug-KS auswählen bzw. eingeben und bestätigen (Schalter = wird rot) unter **Anzeige-> Variable -> Übersicht -> Anzeigen**
- Variableneingabe beenden und im Programm fortfahren, Programm übernimmt die eingegeben Daten (Werkzeug-KS und Fahrzeug-KS)
- Programm startet von der Home-Position und zeichnet das Prueffeld, am Ende faehrt der Roboter wieder auf die Home-Position zurueck (Wird zB eine Linie strichliert eingezeichnet, faehrt der Roboter zwischendurch immer wieder auf die Home-Position zurueck)
- Am Schluss Programmfile abwählen, neues, weiteres Programmfile anwählen, wenn noch weitere Linien eingezeichnet werden sollen – solange bis das Prueffeld fertiggezeichnet ist, wie beschrieben fortfahren

Sollte das Programm in einem „Arbeitsraumfehler“ geraten, so befindet sich das Prüffeld des Fahrzeugs nicht komplett im Arbeitsbereich des Roboters. Entweder Fahrzeug näher am Roboter positionieren oder mit Roboterarm wegfahren und einige Punkte überspringen, dann einen der weiteren Punkte im Programm anwählen und dort fortfahren

Anm.: Singularität

Die Home-/Startposition ist so programmiert, dass der Roboter nicht in eine Singularität gerät (Achsenstopp Achse A4 und A6, Roboter bewegt sich nicht weiter). Sollte es doch einmal vorkommen, ist eine andere Home-Position im Code anzugeben, sodass sich die Achsen im Bereich des Prüffelds frei bewegen können.

Momentane Position:

HOME={AXIS: A1 30,A2 -130,A3 100,A4 100,A5 -20,A6 -20}