Tutorial 33: Using a Round-Robin realtime multi-tasking system

2017-03-05: Tim C. Lueth, MIMED - Technische Universität München, Germany (URL: http://www.mimed.de) - Last Change: 2017-07-07

Contents

Complete List of all Tutorials with Publishable MATLAB Files of this Solid-Geoemtries Toolbox

The following topics are covered an explained in the specific tutorials:

- Tutorial 01: First Steps Using the VLFL-Toolbox for Solid Object Design
- Tutorial 02: Using the VLFL-Toolbox for STL-File Export and Import
- Tutorial 03: Closed 2D Contours and Boolean Operations in 2D
- Tutorial 04: 2½D Design Using Boolean Operators on Closed Polygon Lists (CPL)
- Tutorial 05: Creation, Relative Positioning and Merging of Solid Geometries (SG)
- Tutorial 06: Relative Positioning and Alignment of Solid Geometries (SG)
- Tutorial 07: Rotation of Closed Polygon Lists for Solid Geometry Design
- Tutorial 08: Slicing, Closing, Cutting and Separation of Solid Geometries
- Tutorial 09: Boolean Operations with Solid Geometries
- Tutorial 10: Packaging of Sets of Solid Geometries (SG)
- Tutorial 11: Attaching Coordinates Frames to Create Kinematik Models
- Tutorial 12: Define Robot Kinematics and Detect Collisions
- Tutorial 13: Mounting Faces and Conversion of Blocks into Leightweight-structures
- Tutorial 14: Manipulation Functions for Closed Polygons and Laser Cutting (SVG)
- Tutorial 15: Create a Solid by 2 Closed Polygons
- Tutorial 16: Create Tube-Style Solids by Succeeding Polygons
- Tutorial 17: Filling and Bending of Polygons and Solids
- Tutorial 18: Analyzing and modifying STL files from CSG modeler (Catia)
- Tutorial 19: Creating drawing templates and dimensioning from polygon lines
- Tutorial 20: Programmatically Interface to SimMechanics Multi-Body Toolbox
- Tutorial 21: Programmatically Convert Joints into Drives (SimMechanics)
- Tutorial 22: Adding Simulink Signals to Record Frame Movements
- Tutorial 23: Automatic Creation of a Missing Link and 3D Print of a Complete Model
- Tutorial 24: Automatic Creation of a Joint Limitations
- Tutorial 25: Automatic Creation of Video Titels, Endtitels and Textpages
- Tutorial 26: Create Mechanisms using Universal Planar Links
- Tutorial 27: Fourbar-Linkage: 2 Pose Syntheses and Linkage Export for 3D Printing
- Tutorial 28: Fourbar-Linkage: 3 Pose Syntheses and Linkage Export for 3D Printing
- Tutorial 29: Create a multi body simulation using several mass points
- Tutorial 30: Creating graphical drawings using point, lines, surfaces, frames etc.
- Tutorial 31: Importing 3D Medical DICOM Image Data and converting into 3D Solids
- Tutorial 32: Exchanging Data with a FileMaker Database
- Tutorial 33: Using a Round-Robin realtime multi-tasking system
- Tutorial 34: 2D Projection Images and Camera Coordinate System Reconstruction
- Tutorial 35: Collection of Ideas for Tutorials
- Tutorial 36: Creating a Patient-Individual Arm-Skin Protector-Shell

Motivation for this tutorial: (Originally SolidGeometry 3.5 required)

Matlab can be converted relatively simply to a realtime-multi-tasking environment according to the round-robin method. The following conditions must apply: A) There is a fixed time base for all tasks B) All tasks are called up one after the other in the fixed time clock

The environment presented in this tutorial has the following properties:

- A) Shell character = While the real-time environment is running, Matlab commands can still be typed as before.
- B) All variables of the real-time environment can be modified directly.

List of supported functions

The input-interpreter routine of the real-time environment has additional commands that superimpose comparable commands on the Matlab command line.

- ***HELP*** - shows a help text
- ***EXIT*** or ***QUIT*** - ends the environment
- ***SHOWLIST*** or ***TASKS*** - shows all tasks
- ***KILLTASKS*** or ***KILLALL*** - removes all tasks
- ***ADD*** - appends a task at the task list
- ***KILLLAST*** - removes the last task
- ***BREAK*** or ***STOP*** - stops the realtime execution
- ***STEP*** - runs the realtime loop only ones
- ***GO*** or ***START*** or ***CONT*** - starts the realtime loop
- ***KILL*** Tasknumber - removes a task with that number
- ***SAVE*** - saves the current task list on disk
- ***LOAD*** - loads the current task list on disk
- ***EXE*** or ***EXECUTE*** filename - executes a command line file
- ***edit*** filename edits a command line file
- ***whos*** shows the variables

Intended use of RRrun or RRshell (both are the same)

- ***RRrun*** - starts the Shell
- ***RRrun*** commandstring (lines are spearated by \r)

1. Starting the shell

The following commands could be typed in absolute in the same way as part of the command string. So please try to type them also manually instead of using them als input parameter of RRrun There is no other chance to create a publishable document as to describe them as input parameter.

```
RRrun 'quit'    % Quit immediately
```

```
RRkeyboardLine =
    'quit'
====LOOP STARTS 03-Oct-2023 08:19:08 for 600 SECONDS with CYCLETIME 0.100 SECONDS===========>> END =======
RRrun>>
TERMINATED by User
================LOOP ENDS 03-Oct-2023 08:19:08 USING 0.11 SECONDS ========================
```

```
RRrun 'RRstop=RRcputime+1;'  % Quit after 1 second
```

```
RRkeyboardLine =
    'RRstop=RRcputime+1;'
====LOOP STARTS 03-Oct-2023 08:19:08 for 600 SECONDS with CYCLETIME 0.100 SECONDS===========>> END =======
RRrun>>
================LOOP ENDS 03-Oct-2023 08:19:10 USING 1.20 SECONDS ========================
```

```
RRrun 'LIST \r QUIT' % show the tasks and quit
```

```
RRkeyboardLine =
    'LIST       QUIT'
====LOOP STARTS 03-Oct-2023 08:19:10 for 600 SECONDS with CYCLETIME 0.100 SECONDS===========>> END =======
RRrun>>RRtasklist =
  struct with fields:

       t0: 0.1000
    twarn: 0.1000
    tstop: 1
      cnt: 0
    tlist: []

TERMINATED by User
================LOOP ENDS 03-Oct-2023 08:19:10 USING 0.20 SECONDS ========================
```

```
RRrun 'whos \r QUIT' % show the variables and quit
```

```
RRkeyboardLine =
    'whos       QUIT'
====LOOP STARTS 03-Oct-2023 08:19:11 for 600 SECONDS with CYCLETIME 0.100 SECONDS===========>> END =======
```

```
RRrun>>  Name               Size            Bytes  Class              Attributes

  RRbreak             1x1                 1  logical
  RRcurs              0x0                 0  double             persistent
  RRdelay             1x1                 8  double             global
  RRkeyboardLine      1x6                12  char               global
  RRlastcommand       1x4                 8  char               global
  RRlastexectime      1x1                 8  double
  RRlasttime          1x1                 8  double             global
  RRmaxtime           1x1                 8  double             global
  RRpause             1x1                 8  double
  RRprompt            1x5                10  char               global
  RRstart             1x1                 8  double             global
  RRstop              1x1                 8  double             global
  RRtasklist          1x1               872  struct             global
  RRwindow            1x1                 8  matlab.ui.Figure   global
  RRwindowNr          1x1                 8  double             global
  cmd                 1x4                 8  char
  remain              0x0                 0  char
  token               1x4                 8  char
  varargin            1x1               128  cell


TERMINATED by User
================LOOP ENDS 03-Oct-2023 08:19:11 USING 0.20 SECONDS ========================
```

```
RRrun ('fprintf(''%.2f\n'',RRcputime) \rQUIT') % show the cputime and quit
```

```
RRkeyboardLine =
    'fprintf('%.2f\n',RRcputime)      QUIT'
====LOOP STARTS 03-Oct-2023 08:19:12 for 600 SECONDS with CYCLETIME 0.100 SECONDS==========>> END =======
RRrun>>4.63


TERMINATED by User
================LOOP ENDS 03-Oct-2023 08:19:12 USING 0.20 SECONDS ========================
```

2. Adding realtime tasks and define the stop time

```
RRrun 'ADD fprintf(''%.2f\n'',RRcputime) \r RRstop=RRcputime+1;' % show the cputime every cycle and quit after 1 second
```

```
RRkeyboardLine =
    'ADD fprintf('%.2f\n',RRcputime)       RRstop=RRcputime+1;'
====LOOP STARTS 03-Oct-2023 08:19:12 for 600 SECONDS with CYCLETIME 0.100 SECONDS==========>> END =======
RRrun>>5.41

5.51
5.61
5.71
5.81
5.91
6.01
6.11
6.21
6.31
6.41
6.51
================LOOP ENDS 03-Oct-2023 08:19:14 USING 1.30 SECONDS ========================
```

3. Adding realtime tasks and change the cycle time

```
RRrun 'ADD fprintf(''%.2f\n'',RRcputime) \r RRtasklist.t0=0.05 \r RRstop=RRcputime+1;'
```

```
RRkeyboardLine =
    'ADD fprintf('%.2f\n',RRcputime)       RRtasklist.t0=0.05       RRstop=RRcputime+1;'
====LOOP STARTS 03-Oct-2023 08:19:14 for 600 SECONDS with CYCLETIME 0.100 SECONDS==========>> END =======
RRrun>>7.22
RRtasklist =
  struct with fields:

       t0: 0.0500
    twarn: 0.1000
    tstop: 1
      cnt: 2
    tlist: {' fprintf('%.2f\n',RRcputime)'  [Inf]  [1]}

7.27

7.32
7.37
7.42
```

```
7.47
7.52
7.57
7.62
7.67
7.72
7.77
7.82
7.87
7.92
7.97
8.02
8.07
8.12
8.17
8.22
8.27
8.32
================LOOP ENDS 03-Oct-2023 08:19:15 USING 1.30 SECONDS ========================
```

4. Adding realtime tasks and change the cycle time and kill the task

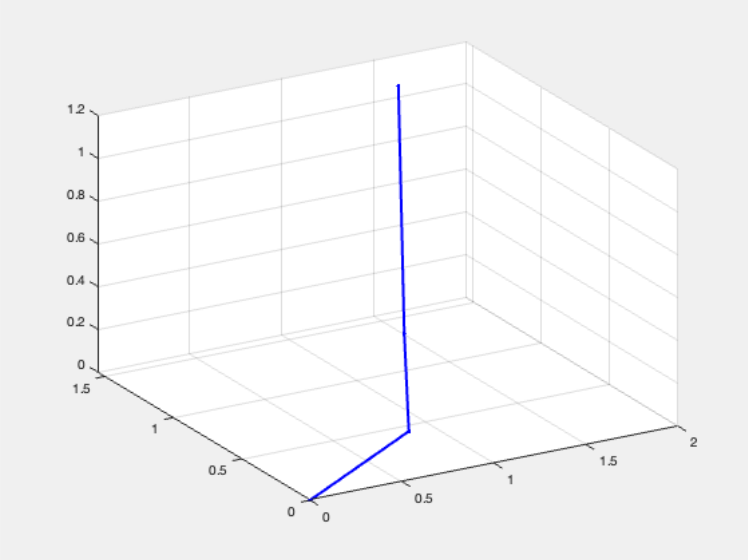
```
RRrun 'ADD fprintf(''%.2f\n'',RRcputime) \r RRtasklist.t0=0.05 \r KILLLAST \r RRstop=RRcputime+1;'
```

```
RRkeyboardLine =
    'ADD fprintf('%.2f\n',RRcputime)       RRtasklist.t0=0.05       KILLLAST       RRstop=RRcputime+1;'
====LOOP STARTS 03-Oct-2023 08:19:16 for 600 SECONDS with CYCLETIME 0.100 SECONDS===========>> END =======
RRrun>>9.01
RRtasklist =
  struct with fields:

       t0: 0.0500
    twarn: 0.1000
    tstop: 1
      cnt: 2
    tlist: {' fprintf('%.2f\n',RRcputime)'  [Inf]  [1]}

9.06
RRtasklist =
  struct with fields:

       t0: 0.0500
    twarn: 0.1000
    tstop: 1
      cnt: 2
    tlist: {0×3 cell}

================LOOP ENDS 03-Oct-2023 08:19:17 USING 1.35 SECONDS ========================
```

5. Run a plotting task and save the task list by using "save"

```
RRrun 'KILLALL \r global PL; PL=[0 0 0]; \r  ADD global PL; PL=[PL; PL(end,:)+rand(1,3)]; \r ADD global PL; delete(gca); VLplot(PL,''b.-'',2); view(-30
```

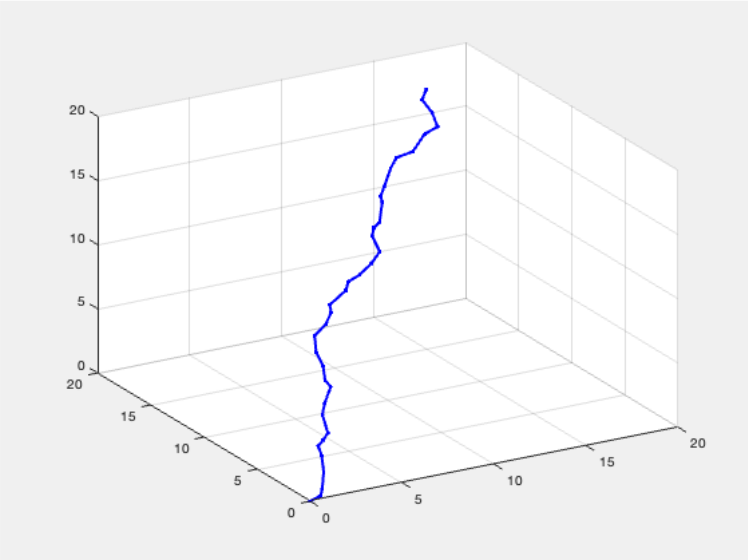
```
RRkeyboardLine =
    'KILLALL       global PL; PL=[0 0 0];        ADD global PL; PL=[PL; PL(end,:)+rand(1,3)];       ADD global PL; delete(gca); VLplot(PL,'b.-',2); vie
====LOOP STARTS 03-Oct-2023 08:19:18 for 600 SECONDS with CYCLETIME 0.100 SECONDS===========>> END =======
RRrun>>RRtasklist =
  struct with fields:

       t0: 0.1000
    twarn: 0.1000
    tstop: 1
      cnt: 0
    tlist: []


RRtasklist saved in RRtasklist.mat
  Name            Size            Bytes  Class     Attributes

  RRtasklist      1x1              1744  struct    global


RRrun: realtime condition broken by 110 milliseconds


RRrun: realtime condition broken by 594 milliseconds

================LOOP ENDS 03-Oct-2023 08:19:22 USING 4.63 SECONDS ========================
```



6. Run the saved task a second time by using "load"

```
RRrun 'load \r start \r copyplot \r RRstop=RRcputime+3;'
```

```
RRkeyboardLine =
    'load       start       copyplot       RRstop=RRcputime+3;'
====LOOP STARTS 03-Oct-2023 08:19:24 for 600 SECONDS with CYCLETIME 0.100 SECONDS==========>> END =======
RRrun>>
New RRtasklist loaded from RRtasklist.mat


RRrun: realtime condition broken by 499 milliseconds

================LOOP ENDS 03-Oct-2023 08:19:28 USING 4.03 SECONDS ========================
```

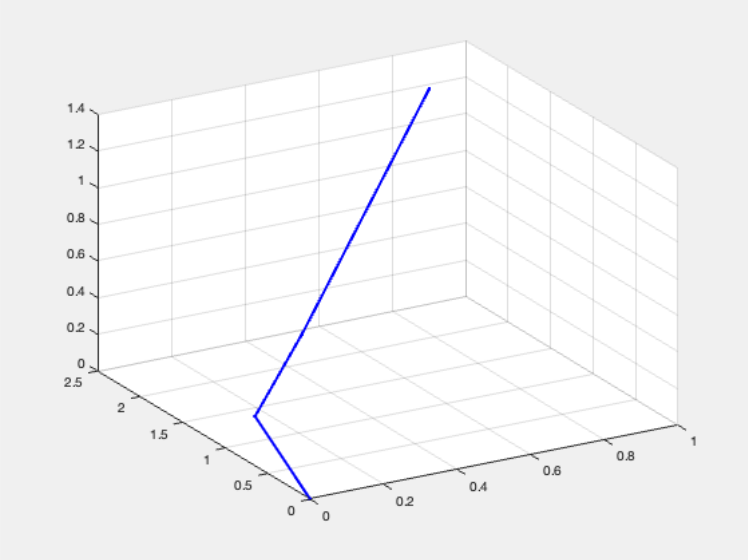


7. Execute a command file

```
RRrun 'execute RRrun_testcommands.txt'
```

```
RRkeyboardLine =
    'execute RRrun_testcommands.txt'
====LOOP STARTS 03-Oct-2023 08:19:28 for 600 SECONDS with CYCLETIME 0.100 SECONDS==========>> END =======
```

```
RRrun>>fname =
    'RRrun_testcommands.txt'
Warning: Inputs must be character vectors, cell arrays of character vectors, or
string arrays.

Error using eval
Unrecognized function or variable 'zuschaurKILLALL'.

RRtasklist =
  struct with fields:

       t0: 0.1000
    twarn: 0.1000
    tstop: 1
      cnt: 4
    tlist: {2×3 cell}
ans =
  2×3 cell array
    {' global PL; PL=[P…'}    {[Inf]}    {[1]}
    {' global PL; delet…'}    {[Inf]}    {[3]}
RRtasklist currently stopped. Use "START" or "STEP" to start task execution.

Elapsed time is 0.017155 seconds.



RRrun: realtime condition broken by 472 milliseconds

RRrun>>================LOOP ENDS 03-Oct-2023 08:19:32 USING 4.01 SECONDS ========================
```



8. Edit a command file

```
edit 'RRrun_testcommands.txt'
```

Final Remarks

```
close all
VLFLlicense
```

```
This VLFL-Lib, Rel.  (2023-Oct-03), is for limited non commercial educational use only!
Licensee: Tim Lueth (Development Version)!
Please contact Tim Lueth, Professor at TU Munich, Germany!
WARNING: This VLFL-Lib (Rel. ) license will exceed at 06-Jul-2078 08:19:33!
Executed 03-Oct-2023 08:19:35 by 'timlueth' on a MACI64 using Mac OSX 13.6 | R2023a Update 5 | SG-Lib 5.4
======================================== Used Matlab products: ========================================
database_toolbox
distrib_computing_toolbox
fixed_point_toolbox
image_toolbox
map_toolbox
matlab
simmechanics
simscape
simulink
=======================================================================================================
```

Published with MATLAB® R2023a