Boot The Bot

Java-based Simulation,
Code Generator and Live Controller
for ABB Robots

by Richard Dank
Institute of Architecture and Media
Graz University of Technology

iam.tugraz.at

Introduction

Abstract

Half a century has passed since the introduction of the first Unimate. Now robots finally have arrived in the minds (and projects) of creatives worldwide. This paper discusses issues concerning algorithmic and innovative offline programming of ABB robots in the context of artistic and architectural purposes. It compares existing software tools for generating RAPID code and introduces *Boot The Bot* – a standalone, cross-platform digital tool to generate or import points plus extra data and export valid RAPID code for immediate execution on ABB robots. This program is the basic software application for several projects and subsequent research themes in TU Graz’s field of expertise *Resource-Efficient Non-Standard Structures*. And it can operate live and online.

Standards?

While harmonized program code gained currency for regular CNC systems, a standard for robot code could not be implemented yet. Industrial robots produced by Unimation Inc. (now owned by Stäubli) for example rely on the Variable Assembly Language (VAL), KUKA on KUKA Robot Language (KRL) and ABB machinery is controlled with RAPID Code. They fundamentally differentiate from each other – not only in terms of semiotics, semantics and syntactics. The interpreters require entirely different datasets to be able to move the robotic arm.

ABB Rapid Code

MoveAbsJ

MoveAbsJ Joints_0000 , v250 , z100 , tool0 ;

…directly rotates the six axes of the robot to a distinct joint position – very common at the beginning and end of any sequence or for ‘untangling’ the robot’s arm. The instruction above moves the flange along a non-linear path to absolute axes positions.

MoveJ

MoveJ Target_0000 , v200 , z0 , weldingPen\WObj:=synthMat0 ;

…is used to move the robot tool to a certain position on the work object not using a straight line – typically used for picking and placing things. In practice this means without simulation one cannot predict the movement of the machine.

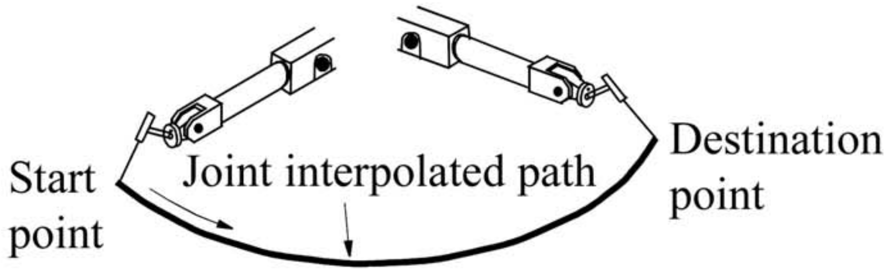
MoveL

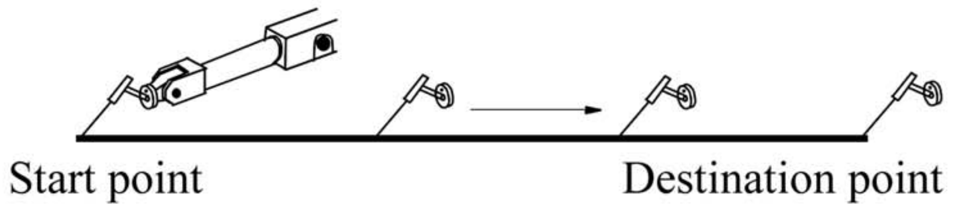
```
MoveL Target_0001 , v100 , z5 , weldingPen\WObj:=synthMat0 ;
```

…is used to move the tip of the tool linearly to a given destination. It is widely used for most operations, still it poses major risk to executable RAPID code. For example none of the six axes can be turned more than 90 degrees within a single instruction.

Motion Instructions

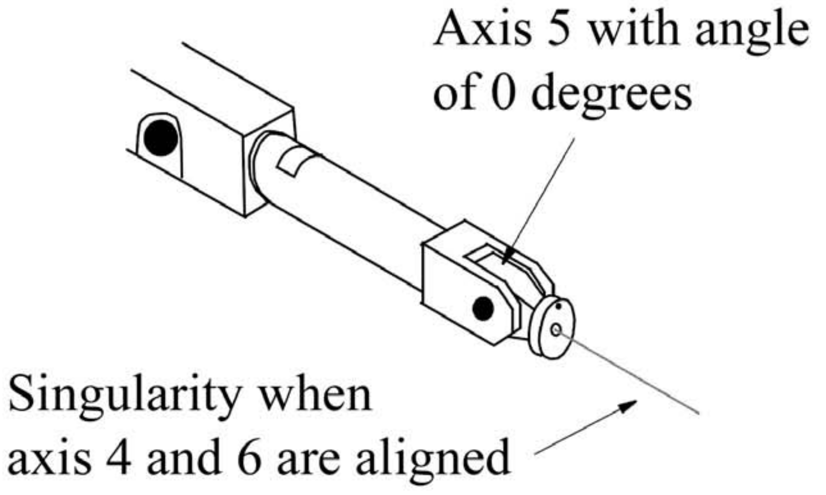
MoveJ is a fast and robust but mostly unpredictable way to move between two points (left). Whereas MoveL describes a linear path, it can run in a number of problems – e.g. the start point and the destination point are too far apart (right).

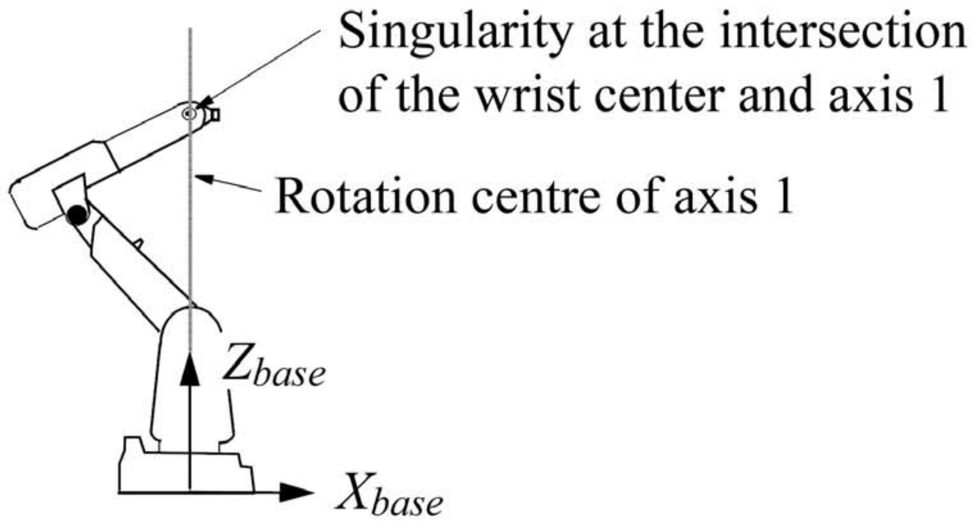




Singularities

Wrist singularities occur when axis 5 is 0 degrees (left). An arm singularity where the wrist center and axis 1 intersect (right).


Axis 5 with angle of 0 degrees
Singularity when axis 4 and 6 are aligned


Singularity at the intersection of the wrist center and axis 1
Rotation centre of axis 1
Z_{base}
X_{base}

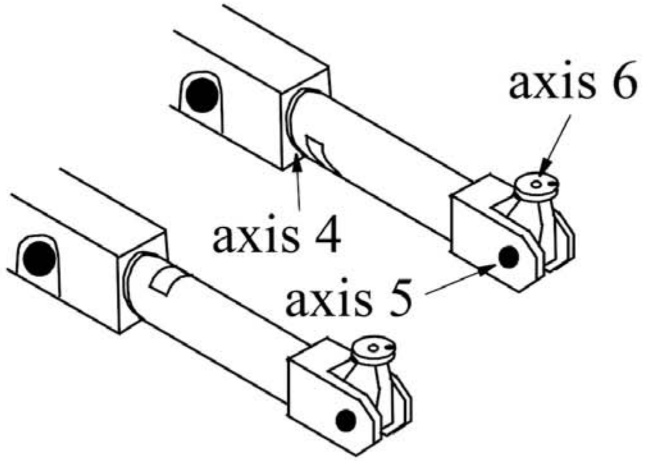
robtarget

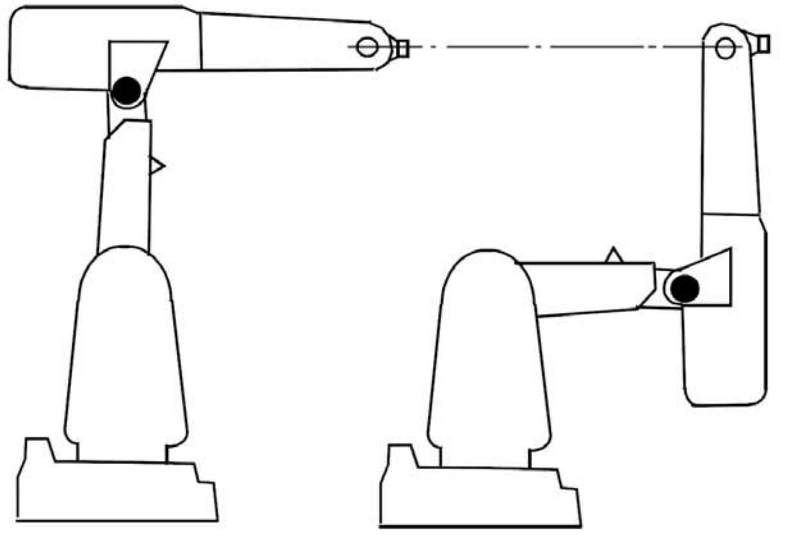
```
CONST robtarget Target_0000 := [
        [257.95,474.5,0.0]
        , [0.060166642,0.010343712,-0.9836818,0.16924272]
        , [1,1,-1,1]
        , [9E9,9E9,9E9,9E9,9E9,9E9]
] ;
```

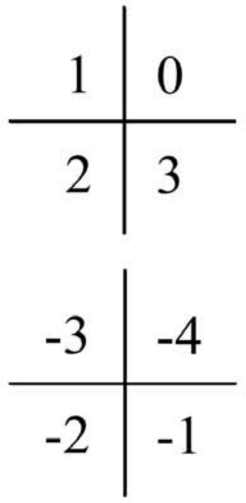
The first set of triple values in box brackets are the X-, Y- and Z-coordinates of the tool tip. The second four values form the orientation of the tool in quaternions, a complex notation of three-dimensional rotations. The last list is reserved for controlling additional axes like linear tracks. However, the third array indeed requires most of the attention. [1,1,-1,1], for example, defines the configuration of the robot. Considering axis 4 and 6 of the robotic arm, it can be done in at least four different configurations.

Configuration

Two different wrist (left) and arm (center) configurations to attain the same position and orientation. Quarter revolutions for the joint angles as final target configuration (right).







Offline Programming

Customized Scripting

Soon after we at the Graz University of Technology decided to acquire ABB robots in 2009/10 it became clear that we needed to **directly control the machines** in order to be able to produce remarkable output. Moreover we agreed that, in the long run, it is valuable and rewarding to be able **to understand and manipulate a machine of that capability on an advanced level** – especially for architects and artists.

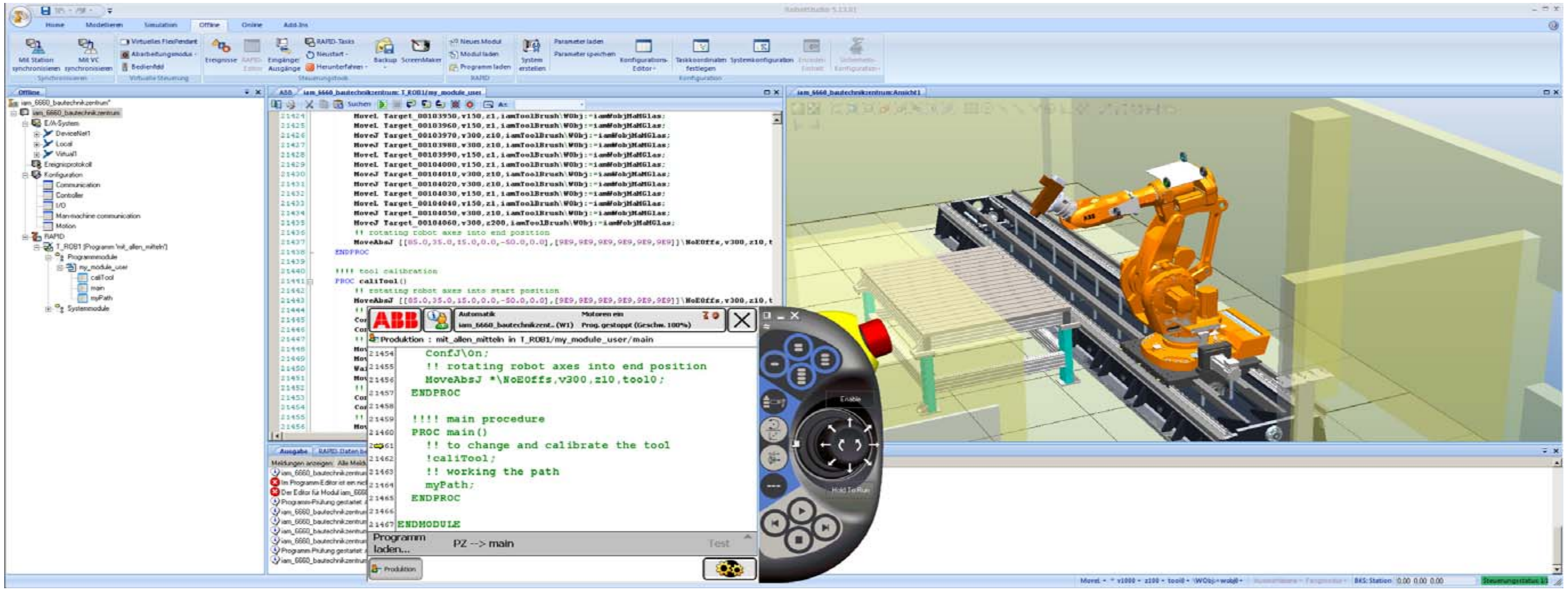
More Bang for the Buck?

Over the last decade design interests have begun shifting away fro
overall forms towards the tessellated worlds of scripted and par
defined structures. The blob may be dead, but the challenge o
geometries in architecture remains. Blob or pattern – advanced co
techniques are becoming indispensable in supporting design, fab

RobotStudio

ABB's very own Software product gives you full access to on- and offline programming. As a result it is a perfect tool for a small number of, especially taught, targets and as a simulator before going into production.

+ The generating of complete programs.
+ Apart from that it is 'the' perfect simulation tool as it is built on the VirtualController.
– It is closed source with no scripting environment. Evidently this means that it is by definition not capable of "fluent bottom-up architectural design […] workflow".
– It needs to completely simulate the robotic arm's motion in order to find out any problems with the configuration plus the user must again solve them by hand.



```
        MoveL Target_00103950,v150,z1,iamToolBrush\WObj:=iamWobjMaMGlas;
        MoveL Target_00103960,v150,z1,iamToolBrush\WObj:=iamWobjMaMGlas;
        MoveJ Target_00103970,v300,z10,iamToolBrush\WObj:=iamWobjMaMGlas;
        MoveJ Target_00103980,v300,z10,iamToolBrush\WObj:=iamWobjMaMGlas;
        MoveL Target_00103990,v150,z1,iamToolBrush\WObj:=iamWobjMaMGlas;
        MoveL Target_00104000,v150,z1,iamToolBrush\WObj:=iamWobjMaMGlas;
        MoveJ Target_00104010,v300,z10,iamToolBrush\WObj:=iamWobjMaMGlas;
        MoveJ Target_00104020,v300,z10,iamToolBrush\WObj:=iamWobjMaMGlas;
        MoveL Target_00104030,v150,z1,iamToolBrush\WObj:=iamWobjMaMGlas;
        MoveL Target_00104040,v150,z1,iamToolBrush\WObj:=iamWobjMaMGlas;
        MoveJ Target_00104050,v300,z10,iamToolBrush\WObj:=iamWobjMaMGlas;
        MoveJ Target_00104060,v300,z200,iamToolBrush\WObj:=iamWobjMaMGlas;
        !! rotating robot axes into end position
        MoveAbsJ [[85.0,35.0,15.0,0.0,-50.0,0.0],[9E9,9E9,9E9,9E9,9E9,9E9]]\NoEOffs,v300,z10,t
    ENDPROC

    !!!! tool calibration
    PROC caliTool()
        !! rotating robot axes into start position
        MoveAbsJ [[85.0,35.0,15.0,0.0,-50.0,0.0],[9E9,9E9,9E9,9E9,9E9,9E9]]\NoEOffs,v300,z10,t
```

Automatik Motoren ein
iam_6660_bautechnikzent.. (W1) Prog. gestoppt (Geschw. 100%)
Produktion : mit_allen_mitteln in T_ROB1/my_module_user/main

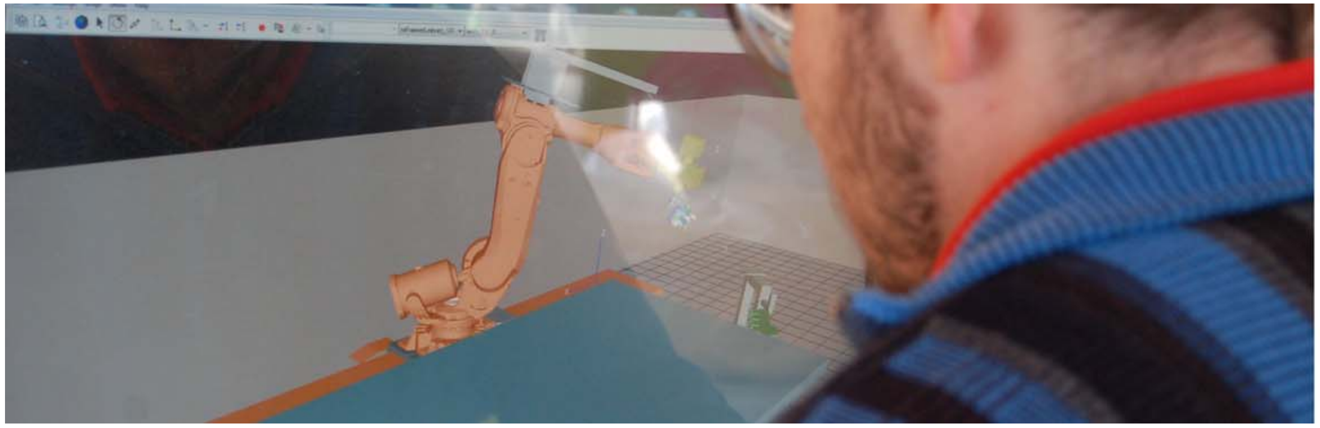
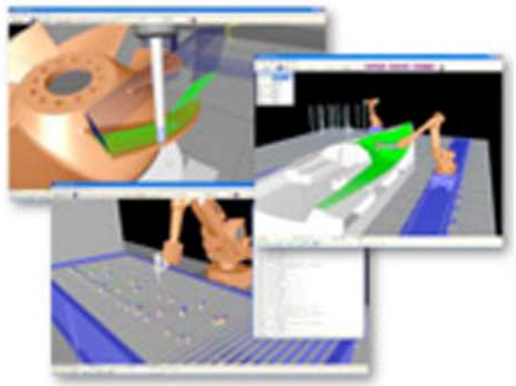
```
    ConfJ\On;
    !! rotating robot axes into end position
    MoveAbsJ *\NoEOffs,v300,z10,tool0;
  ENDPROC

  !!!! main procedure
  PROC main()
    !! to change and calibrate the tool
    !caliTool;
    !! working the path
    myPath;
  ENDPROC

ENDMODULE
```

Programm laden... PZ --> main Test

Pi-Path

An easy yet powerful tool to convert three or five axes CNC code into multi-axes robot programs. It is perfect for milling procedures, can be ‘abused’ for other tasks, but is not able to parse elaborate code like different motion instructions etc.

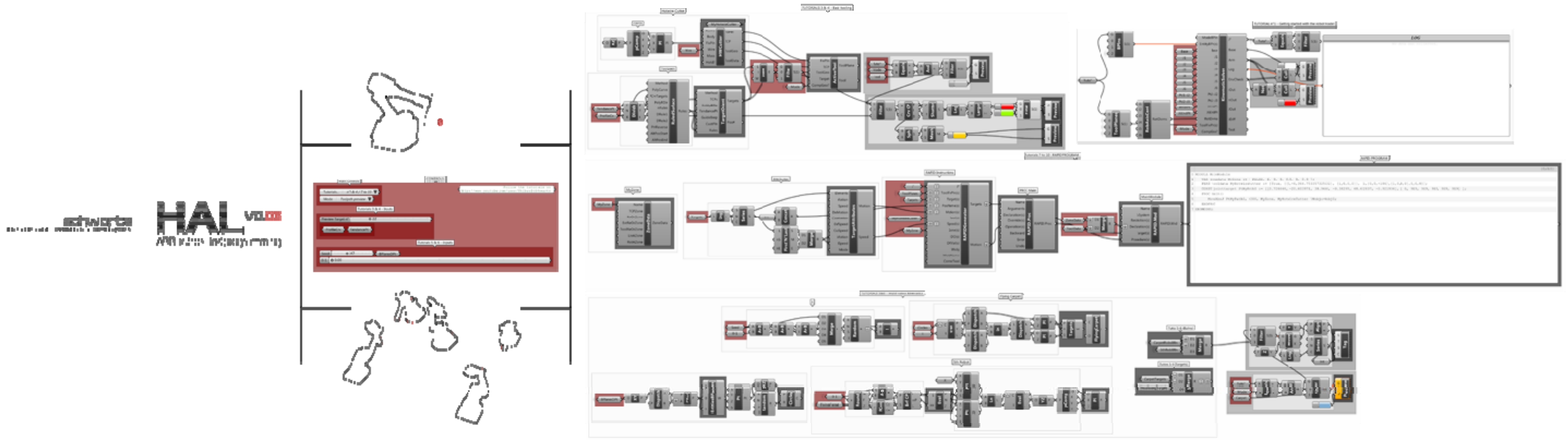
+ Utra-fast in writing the configuration data.
– As it is basically meant to translate pre-computed APT-files, it needs manual input on errors as well.
– Cannot be programmed.

HAL

Alike robotsinarchitecture's Parametric Robot Control (PRC) for KUKA, Thibault Schwartz's HAL is a plug-in for Rhino's generative modeler Grasshopper. It is sophisticated enough for ingenious applications, still straightforward so that even inexperienced users can work with it. Definitely a recommendation.

- + HAL simulates and produces code for points well that are close to each other.
- – But it doesn't take configuration problems or singularities into account – unless you solve them manually.
- – Apart from that it assumes that there is only one viable configuration for one target, which again could make things harder.
- – Chained to Windows, Rhino and Grasshopper.



Boot The Bot

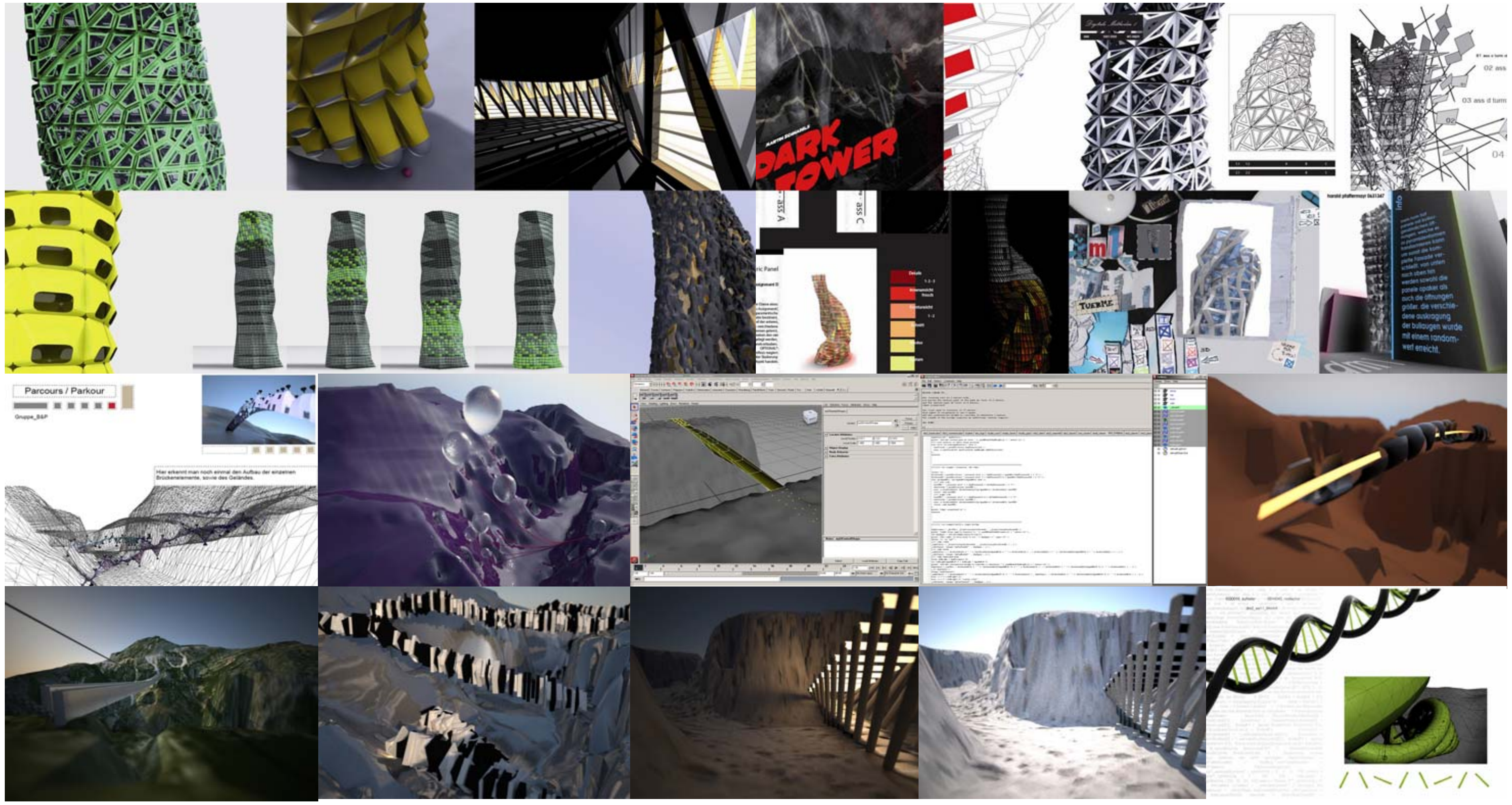
Open Source

We wanted the application to be as open as possible. Thus it is written under the creative common license in **Java with the implemented integration into Processing** – everything open source.

We deliberately did not fall back on a CAD backbone like Maya/MEL or Rhino/Grasshopper and of course we had to take a loss there. Designers that are not willing and/or able to calculate the initial points of the motion paths in 3D will not be able to use the full potential.

DM2

But architecture students at the TUG have the mandatory class Digitale Methoden der Gestaltung in their third semester. In DM2 they take **their first steps into contemporary algorithmic design methods.**



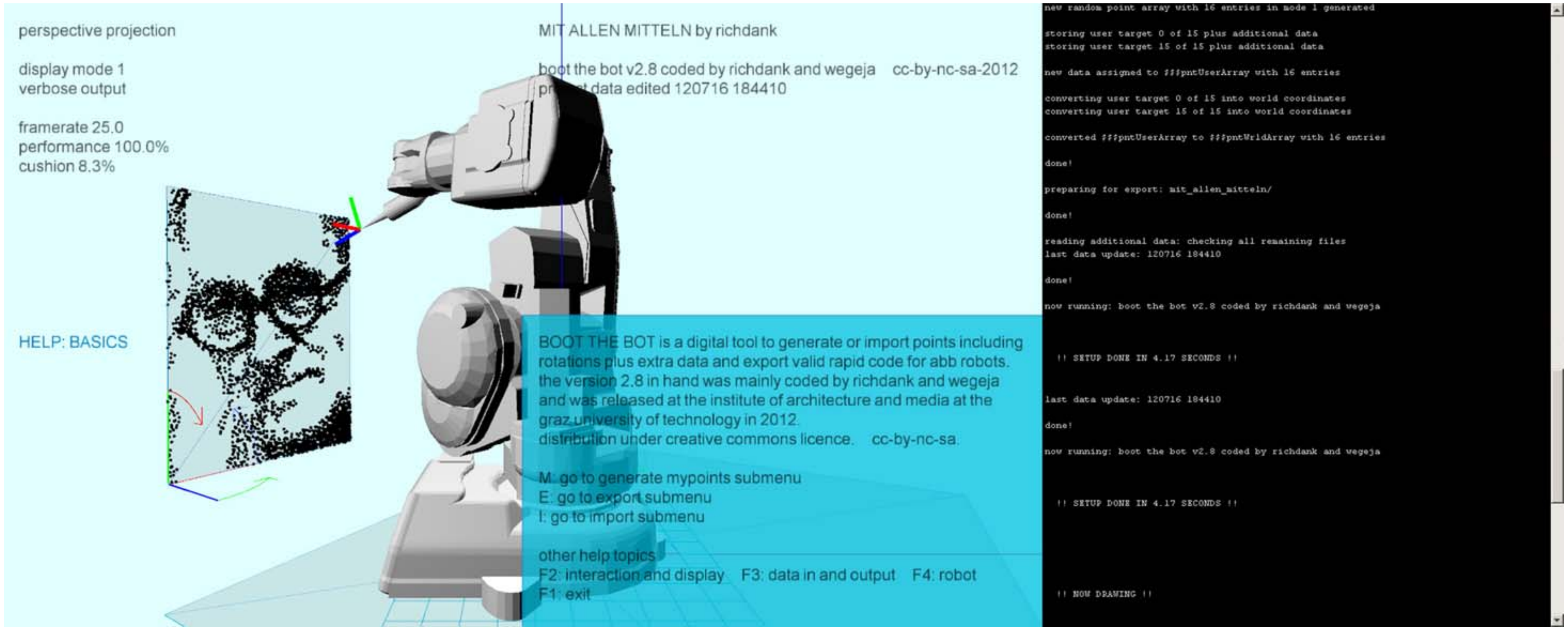
Connectable,
Interactive

Thus we were able to open BTB to **all software platforms**.

Moreover, by using Java we made it possible that projects could easily **access live and/or online input.** Another interesting aspect of BTB is that data, like the measured work objects or tools on the real robot, can be read and interpreted. So you can make declarations both ways – either generate them or grab existing ones..

AutoConf

However the key feature is the **automatic configuration of the targets**. Multiple possible configurations are calculated and due to several parameters BTB then decides which is best. If there is still no satisfying solution, the motion path is interpolated and additional crucial targets are added. Even singularities can be avoided to a certain extend. If possible the robot just moves around them.



```
perspective projection

display mode 1
verbose output

framerate 25.0
performance 100.0%
cushion 8.3%

HELP: BASICS
```

```
MIT ALLEN MITTELN by richdank

boot the bot v2.8 coded by richdank and wegeja     cc-by-nc-sa-2012
pr[illegible] data edited 120716 184410
```

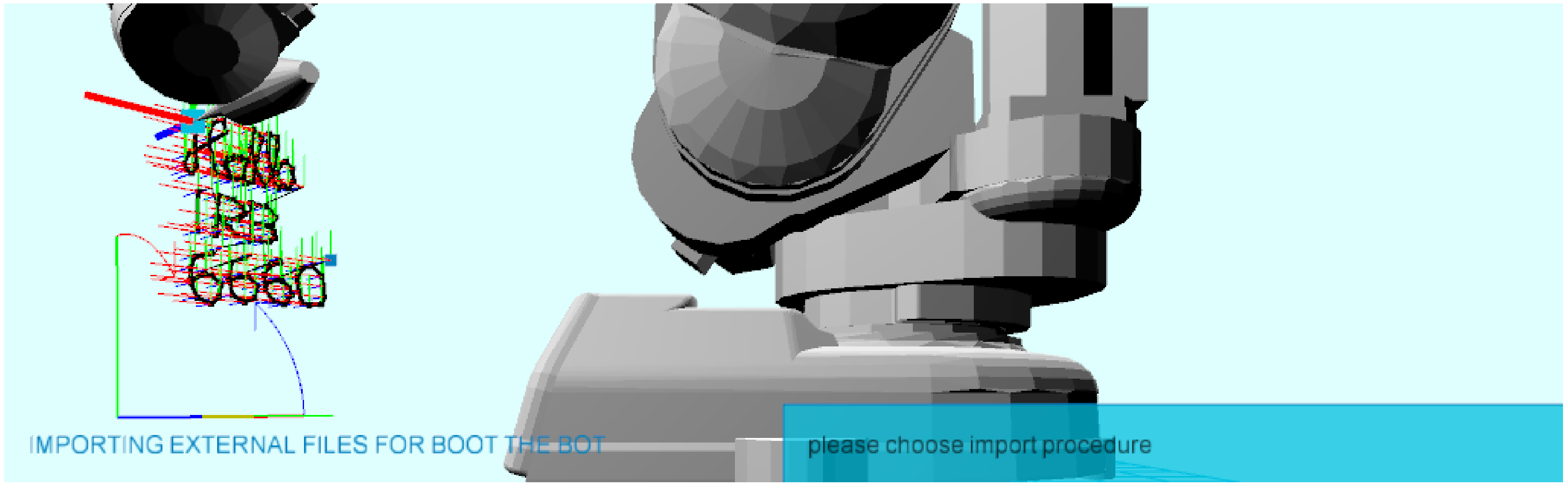
```
BOOT THE BOT is a digital tool to generate or import points including
rotations plus extra data and export valid rapid code for abb robots.
the version 2.8 in hand was mainly coded by richdank and wegeja
and was released at the institute of architecture and media at the
graz university of technology in 2012.
distribution under creative commons licence.     cc-by-nc-sa.

M: go to generate mypoints submenu
E: go to export submenu
I: go to import submenu

other help topics
F2: interaction and display     F3: data in and output     F4: robot
F1: exit
```

```
new random point array with 16 entries in mode 1 generated

storing user target 0 of 15 plus additional data
storing user target 15 of 15 plus additional data

new data assigned to ###pntUserArray with 16 entries

converting user target 0 of 15 into world coordinates
converting user target 15 of 15 into world coordinates

converted ###pntUserArray to ###pntWrldArray with 16 entries

done!

preparing for export: mit_allen_mitteln/

done!

reading additional data: checking all remaining files
last data update: 120716 184410

done!

now running: boot the bot v2.8 coded by richdank and wegeja



  !! SETUP DONE IN 4.17 SECONDS !!


last data update: 120716 184410

done!

now running: boot the bot v2.8 coded by richdank and wegeja



  !! SETUP DONE IN 4.17 SECONDS !!




  !! NOW DRAWING !!
```

Postprocessor

In summary, there are two ways of utilizing BTB. First of all one can just take it as a **multi-platform postprocessor for converting target information into RAPID code** to address ABB robots. The environment allows for

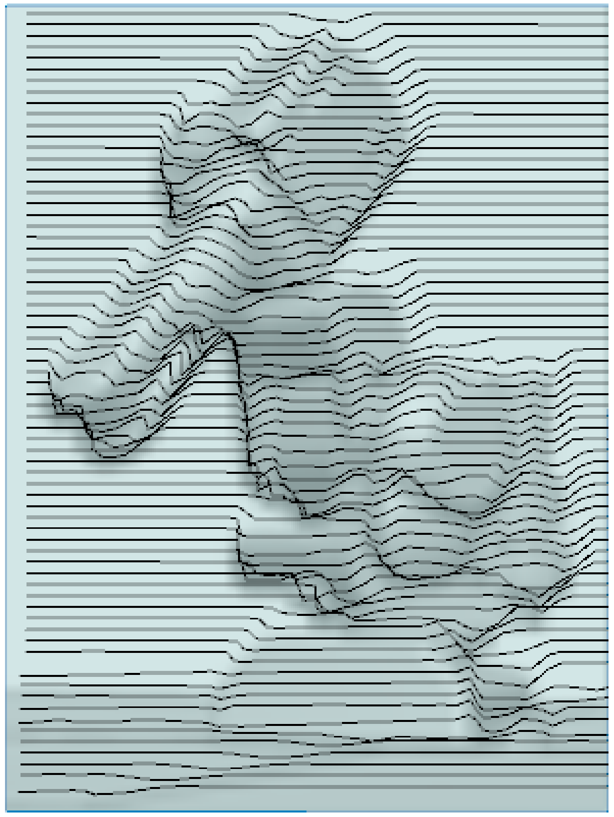
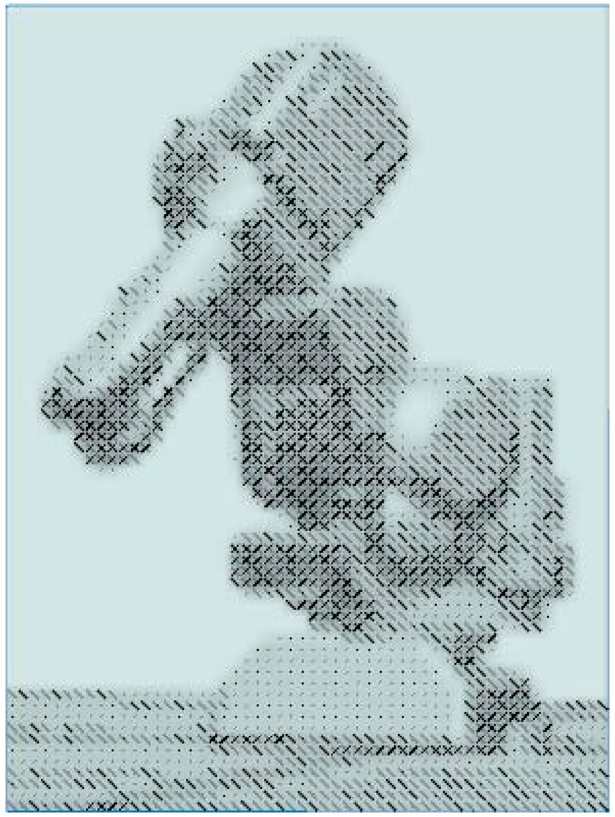
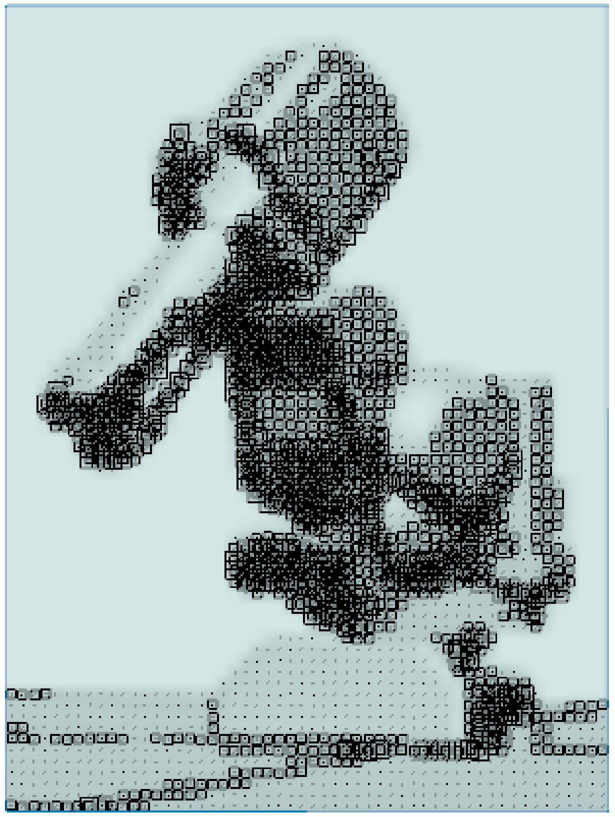
+ **Importing** a variety of CAD-formats and plain-text data delineating motion paths.
+ Different visual and textual screen displaying modes, orthographic and perspective views for debugging.
+ Additional positioning, motion and timing settings.
+ Tool and work object import and export.
+ **Full simulation** of the complete workflow.
+ **Export working ABB RAPID code.**
+ Direct physical control over the robot via network connection.


IMPORTING EXTERNAL FILES FOR BOOT THE BOT
please choose import procedure

Standalone

Nonetheless, **the intrinsic intention of BTB is the standalone version**. Additionally it enables the user to

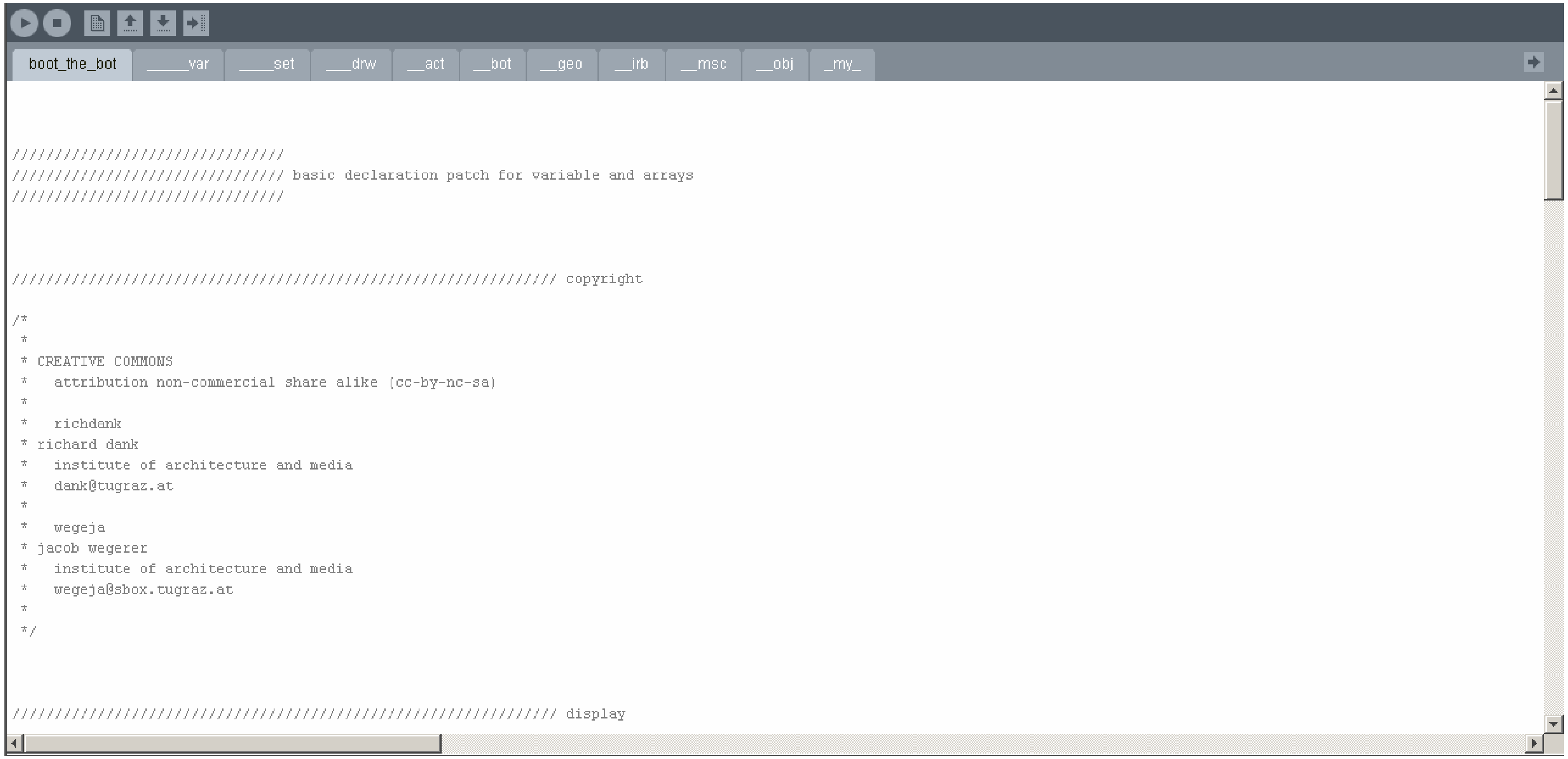
+ Compute the robot’s motion by **self-written functions** completely within BTB.
+ **Grab live input** and/or interact with the output.
+ **Generate** (convert data), **verify** (simulate the sequence) **and finalize** (write RAPID code) **all in one non-linear procedure.**
+ Run it locally or on the web.

Adapt BTB

And again I need to mention the fact that everything is open source. So if someone is not happy with the automatic configuration parameters, the interpolation steps, the interpretation of fragmentary motion data or even the colors

+ Everyone can adapt the code for their own needs.

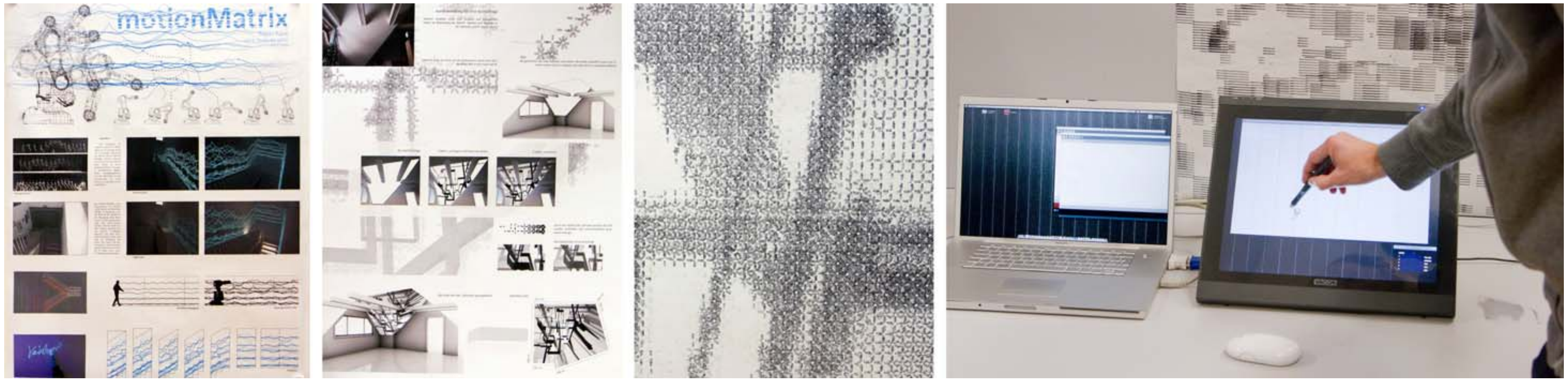


```
//////////////////////////////////
////////////////////////////////// basic declaration patch for variable and arrays
//////////////////////////////////



////////////////////////////////////////////////////////////////////// copyright

/*
 *
 * CREATIVE COMMONS
 *   attribution non-commercial share alike (cc-by-nc-sa)
 *
 *   richdank
 * richard dank
 *   institute of architecture and media
 *   dank@tugraz.at
 *
 *   wegeja
 * jacob wegerer
 *   institute of architecture and media
 *   wegeja@sbox.tugraz.at
 *
 */



////////////////////////////////////////////////////////////////////// display
```

Projects using BTB

papier peint

Project posters of *motionMATRIX* by Marvi Basha and *Raumverfremdung* by Kathrin Hiebler, the image of the final result on the roof truss corner and Stefan Höll’s *Individual Wallpaper* manipulated on a Wacom board (from left to right).

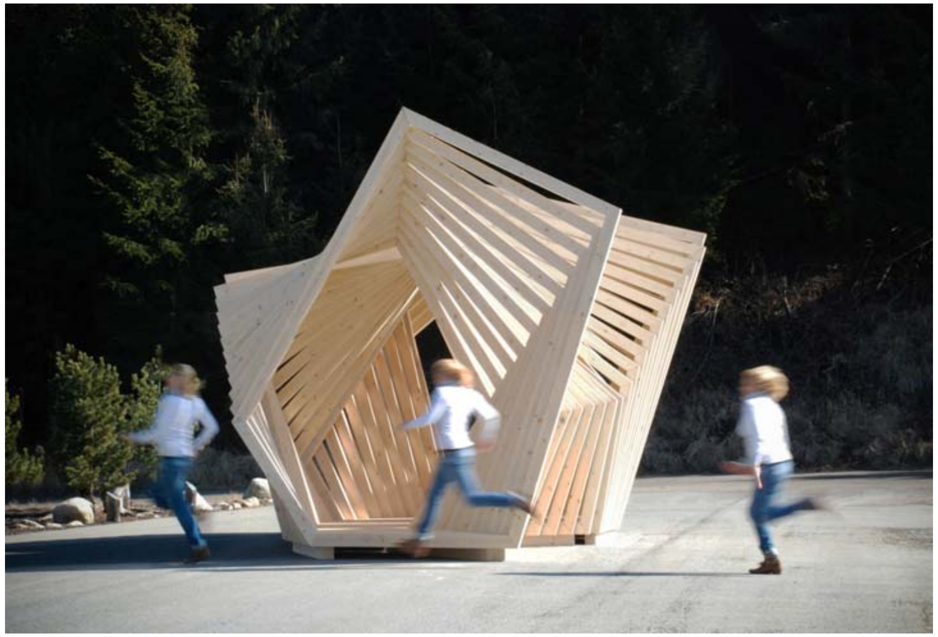
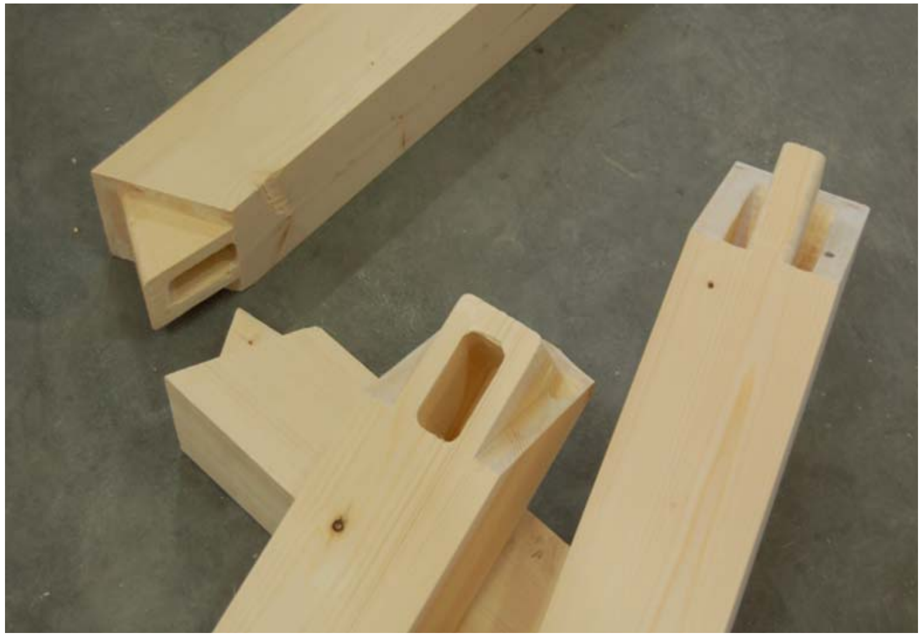


papier peint

Peter Kaufmann / Robert Schmid’s ***Die Anwesenheit der Abwesenden und Hirsch*** bleached deer on a couch, their and Paul Pritz’s tool, extracts from the robot-produced ***[SYN]these*** and Simone Mayr’s guidance system *Ohne Worte* (from left to right).

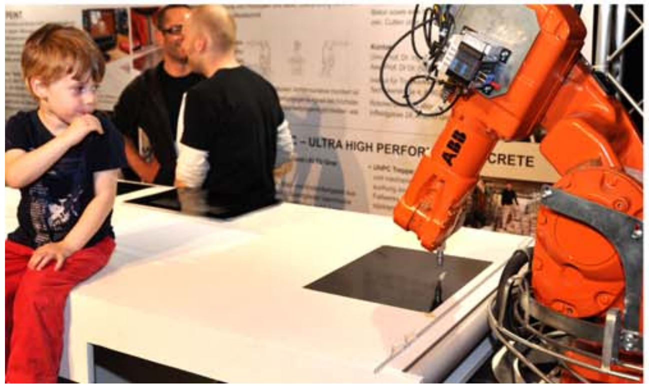
The Framed Pavilion

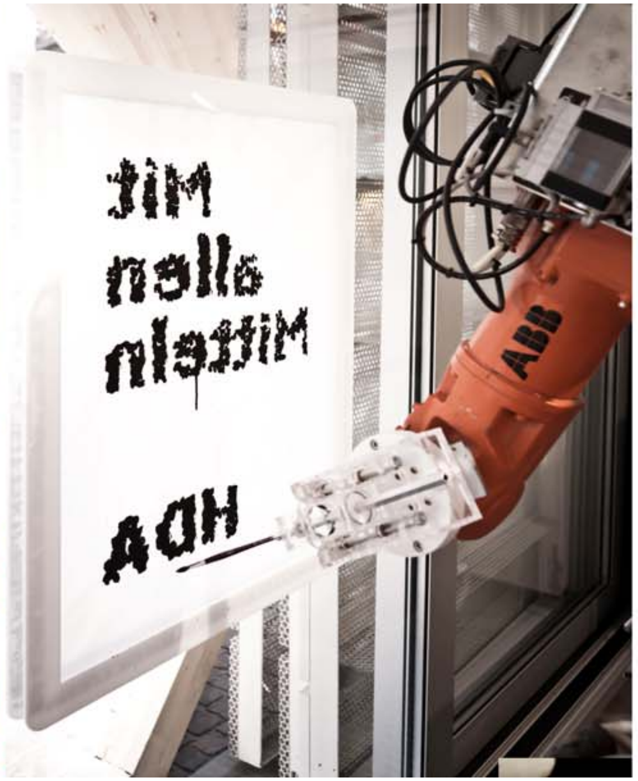
The Objective: Design a structure and all joints **solely made from timber**, no glue or other fasteners or fixings allowed. For the realization **use the capabilities of a 6-axes robot** on an additional linear axis. Moreover **the entire project must be applied parametrically!**

By all means

Two Pictures from TUG’s 200 years anniversary, the *China Ink Painting Robot* plus Wolfgang Tschapeller, Peter Cook and Marjan Colletti picking up their portraits at the HDA (from left to right).





Conclusion and Outlook

Seamless Incorporation

“An understanding of digital design as a unique set of design logic demands a formulation of the symbiosis between the product of design and the way it is now conceived, generated and materialized in digital media.”

Reframing Rivka Oxman’s statement, BTB provides this conjunction between the most open way of formalizing an idea – alphanumeric code – and the most versatile machine available – the robot.

Acknowledgement

The evolution of Boot The Bot started with stirring stimulus at IAM and energetic support of ABB – especially by Michael Roth. Apart from him Richard Dank certainly needs to give thanks to Jacob Wegerer. His contribution and collaboration exceeded his engagement as teaching assistant by far. And finally he would like to lay great stress upon the inspiring efforts and wonderful projects of all the peers and students that used BTB ever since it came out.