# Force Controlled Assembly of Flexible Aircraft Structure

Stolt, Andreas; Linderoth, Magnus; Robertsson, Anders; Jonsson, Marie; Murray, Tom

2011

2011 IEEE International Conference on Robotics and Automation
Shanghai International Conference Center
May 9-13, 2011, Shanghai, China

# Force Controlled Assembly of Flexible Aircraft Structure

Andreas Stolt, Magnus Linderoth, Anders Robertsson, Marie Jonsson, and Thomas Murray

*Abstract*— **The use of industrial robots in the aircraft industry has been hampered by a combination of poor accuracy of the robots and poor calibration of the workcell, and also manufacturing variability in composite parts. A way to handle these difficulties is using force control. An experimental case where a semi-compliant rib is aligned to multiple surfaces is used as an example to show this. The constraint-based task specification framework is used for the modelling and control, and the search and alignment sequence required for the assembly is modeled with a state machine. An implementation on an industrial robot system is presented and experimental data is evaluated. The described approach is easy to apply to other fields and more complicated assembly operations as well.**

## I. INTRODUCTION

Current automation in the aircraft industry is usually not especially easy to modify, as the tolerances and demands are tight. One way to change this would be to introduce more industrial robots. The control of them can however not be done in the traditional way with position control, as the accuracy of the robot and the calibration of the workcell is generally not good enough. As robots are flexible structures the accuracy is even worse when external forces are present. One way to handle this problem is using force control, if it is possible to specify desired forces instead of positions. Another problem in the industry is manufacturing variability, which an automated system will have to be able to handle. In cases where the solution to the variability is to make the pieces fit as good as possible force control is a well suited alternative.

A previous application of force control in the aircraft industry is described in [7]. The application is drilling and force control is used to avoid sliding movements, to get accurate holes.

The assembly case in this paper is to ”best fit” a semi-compliant rib to multiple surfaces using a search and alignment sequence, previously described in [6] from a production/assembly technology point of view, whereas this paper is focused on the task control. The full-size rib is made of carbon fiber, but a smaller metal replica has been used for the experiments. The experimental jig consists of brackets in a metal frame, which can be moved to make it possible to replicate variability. The robot system used is an ABB IRB2400/16 with a robot system controller extended for external sensor integration [2], [3]. See Fig. 1 for a photo of the setup.

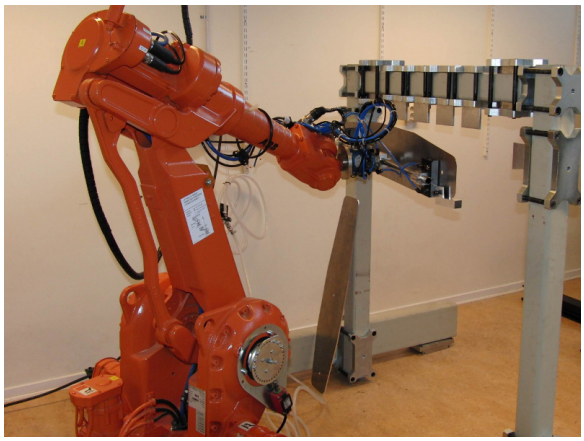

Fig. 1: The robot gripping the rib in front of the experimental jig.

The task has been implemented using the constraint-based task specification methodology [5]. It is a general framework that makes it easy to algorithmically incorporate multiple sensors, geometric uncertainties and to handle both redundant manipulators and redundant tasks. The task specification is also easy to use and to reuse. A previous implementation is described in [8], which is based on the OROCOS project [4].

## II. TASK MODELLING

The constraint-based task specification framework defines the relative motion of objects by imposing constraints, such as position or force constraints. To be able to specify these a so called kinematic chain is needed, which consists of two object frames and two feature frames. The first object frame is usually attached to the object one wants to manipulate and the second object frame is usually attached to the robot. The feature frames should be attached to features on the object to manipulate and on the robot. There might be several possibilities for this modelling, the feature frames should therefore be chosen in such a way that the task constraints of the desired motion become as easy as possible to specify. A kinematic chain should have 6 degrees of freedom, and they are distributed over the transformations between the feature and the object frames. These six degrees of freedom are represented by $\chi_f$, the so called feature coordinates. They can further on be divided into the degrees of freedom between each of the defined object and feature frames.

Aside from the feature coordinates there might also be

Andreas Stolt, andreas.stolt@control.lth.se, Magnus Linderoth, and Anders Robertsson, are with the Department of Automatic Control, LTH, Lund University, Sweden.

Marie Jonsson, and Thomas Murray are with the Department of Assembly Technology, Linköping University, Sweden

This work has partly been made within the project Productive Flexible Automation (ProFlexA) under the SSF/ProViking-grant V08.05

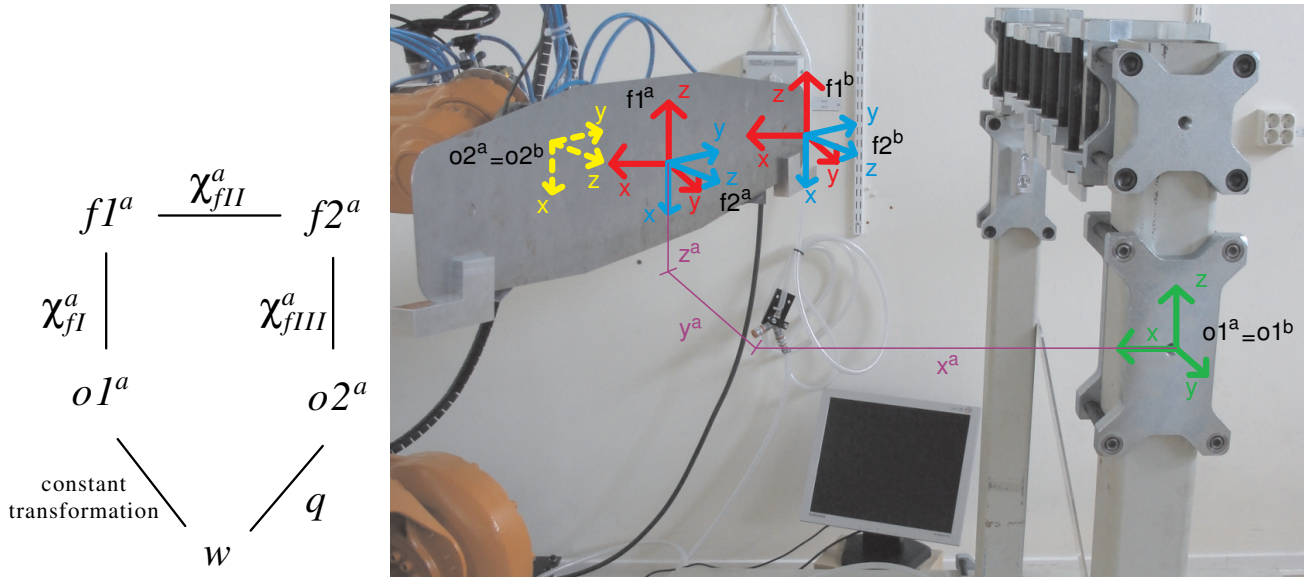


Fig. 2: Illustration of the different coordinate frames in the assembly task. Left: Schematic picture of loop $a$. The feature system is here the chain between $o1$ and $o2$. $w$ denotes the world coordinate system and $q$ denotes the robot joint coordinates. Right: Experimental setup with the two kinematic loops in the task description. Three of the feature coordinates are also illustrated.

uncertainties in the pose between the previously defined coordinate frames. To handle this an extra transformation between each of the the previously mentioned coordinate frames is introduced, and the degrees of freedom in these transformations are denoted $\chi_u$, the uncertainty coordinates.

The variables one wants to constrain are chosen by specifying outputs $y$. Each output can in general be a function of the feature and the robot joint coordinates, but if the kinematic chains have been chosen properly the outputs will in most cases directly correspond to some of the feature coordinates.

The kinematic chain can be seen as a virtual robot where the feature coordinates are the joint variables. Each such virtual robot is in the sequel called a feature system.

### A. Kinematic chains in the specific task

Two kinematic chains are used in the assembly task. The first one is called loop $a$ and the second loop $b$. An illustration of them and their coordinate frames is found in Fig. 2.

- Object frame $o1^a$ is fixed to the jig, and it is connected to the robot base, the world coordinate frame, by a constant transformation. Object frame $o1^b$ coincides with $o1^a$.
- Feature frame $f1^a$ has the same orientation as $o1^a$, but has its origin on the center point of the rib that is grasped by the robot. Feature frame $f1^b$ is similarly defined, but it has its origin on the leftmost point of the rib.
- Feature frame $f2^a$ has its origin in the same point as $f1^a$, but its orientation is the same as the robot flange frame. The same holds for feature frame $f2^b$ and $f1^b$, respectively.
- Object frame $o2^a$ coincides with the flange of the robot, as well as object frame $o2^b$.

The corresponding feature coordinates used are (I refers to coordinates between $o1$ and $f1$, II coordinates between $f1$ and $f2$ and III coordinates between $f2$ and $o2$)

$$\chi^a_{fI} = (x^a, y^a, z^a) \qquad \chi^b_{fI} = (x^b, y^b, z^b)$$
$$\chi^a_{fII} = (\varphi^a, \theta^a, \psi^a) \qquad \chi^b_{fII} = (\varphi^b, \theta^b, \psi^b)$$
$$\chi^a_{fIII} = (-) \qquad \chi^b_{fIII} = (-)$$

where $x^a$, $y^a$, $z^a$, $x^b$, $y^b$ and $z^b$ are expressed in $o1^a$ and $o1^b$ respectively. The rest are expressed in $f1^a$ and $f1^b$. The angle coordinates are Euler ZYZ-angles. The feature coordinates are then collected into

$$\chi^a_f = \left(\chi^a_{fI}, \chi^a_{fII}, \chi^a_{fIII}\right) \qquad \chi^b_f = \left(\chi^b_{fI}, \chi^b_{fII}, \chi^b_{fIII}\right)$$

In this task all feature coordinates are chosen as outputs, according to

$$y_1 = x^a \qquad y_7 = x^b$$
$$y_2 = y^a \qquad y_8 = y^b$$
$$y_3 = z^a \qquad y_9 = z^b$$
$$y_4 = \varphi^a \qquad y_{10} = \varphi^b$$
$$y_5 = \theta^a \qquad y_{11} = \theta^b$$
$$y_6 = \psi^a \qquad y_{12} = \psi^b$$

The exact position and orientation of the jig in relation to the robot, as well as the exact grasp of the rib on the robot are not assumed to be perfectly known in this task. These uncertainties are however chosen not to be explicitly modeled as uncertainty coordinates, but are handled via *guarded search motions* in the assembly sequence, i.e. as motions 'guarded' by sensor measurements to e.g., determine exact position of making contact.

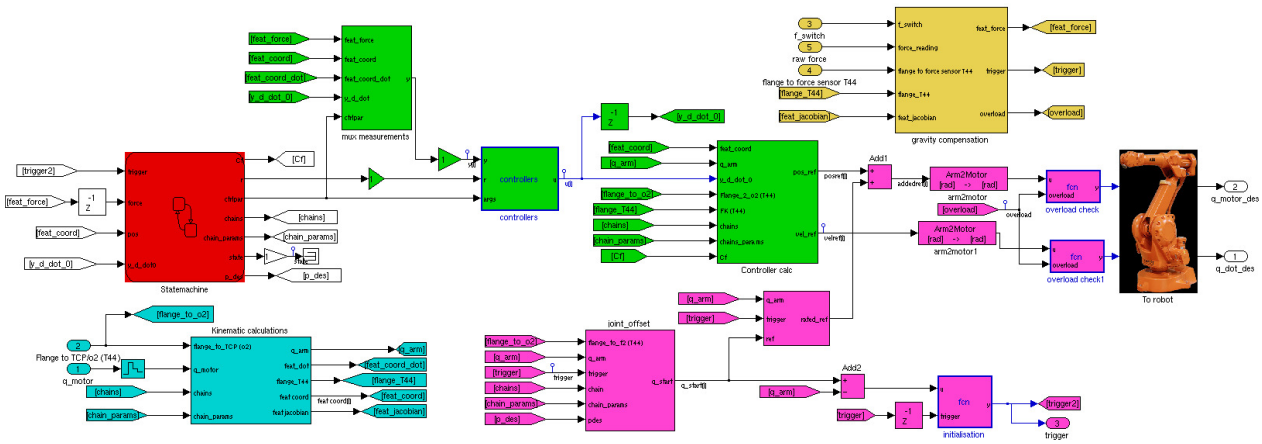


Fig. 3: The simulink model used for code generation containing the control sequences and the control algorithm.

### B. Sensors

The only external sensor used in the task is a six degree of freedom JR3 force/torque sensor, which is mounted on the wrist of the robot. The measured forces in the sensor coordinate frame are transformed to the feature coordinate system using equation (1), where $J$ is the Jacobian of the feature system. The contact point is assumed to be known.

$$F_{feat} = J^T F_{sens} \tag{1}$$

Before the transformation a static gravity compensation and a rigid force transmission from the sensor coordinate system to the robot flange has also been performed.

### C. Control

A velocity based control scheme is used, where it is assumed that the robot is an ideal velocity controlled system. Velocities of the outputs $y$ are specified according to (2). The first term represents feed-forward and the second feedback. The reference value $r$ and the measurement $y_{meas}$ might represent both position, velocity and force depending of which type of controller $C$ that is used.

$$\dot{y}_d^0 = \dot{y}_d + C(r, y_{meas}) \tag{2}$$

Three different controllers $C$ are used; PI position and velocity controllers, and an impedance controller. The implementation of the PI controllers incorporate anti-windup functionality as described in e.g., [1]. The impedance controller is used for force control and uses the algorithm (3) for calculating the desired acceleration. This is then integrated and used as control signal. The parameter $M$ can be interpreted as the mass the robot should have; a larger $M$ makes the controller react less to control errors. The other parameter $D$ is the damping.

$$\ddot{x}_{des} = \frac{1}{M}\left(F - F_{ref} - D\dot{x}\right) \tag{3}$$

The feed-forward part is used only for velocity control, i.e. search operations, and the feedback part mainly for position and force control.

## III. SYSTEM

The workcell and experimental setup consists of an ABB IRB2400/16 industrial manipulator together with the S4CPlus robot control system. This has been extended with an open robot control system [2], [3], which allows for manipulation of position, velocity and torque references. Additional hardware is used to handle the force sensor, and the data from it is sent to the robot program via a network connection. The flexible rib is grasped by the robot tool consisting of a supporting frame with vacuum grippers, se Fig. 1.

The controller is built in Mathwork's Matlab/Simulink environment, with the sequencing in Stateflow. Executable code is then generated with the Real-Time Workshop toolbox. This code is downloaded to and run from the robot controller.

## IV. SOFTWARE IMPLEMENTATION

The software is implemented using Mathwork's Matlab/Simulink/Stateflow environment. The program is fed with measurements from the robot and the available sensors. The output from the program is robot joint position and velocity references for the low-level controller loops. The program is divided into two parts; one that is specific for the current task and the rest is independent of it. A screenshot of the used Simulink model is found in Fig. 3. The red box labelled Statemachine is the task dependent part, and it is a Stateflow chart that contains the assembly sequence. All parameters and constraints are set in this block and sent to the rest of the program. The other part of the software contains several subparts. The yellow block contain gravity compensation and force transformation, the blue block kinematic calculations, the green blocks control algorithm and the pink ones start-up procedure and conversion from arm-angles to robot motor-angles.

The kinematic calculations include forward and inverse kinematics for both the feature and the robot systems, as well as calculation of the Jacobians. All calculations are numeric and are made at each sampling instant (4 ms).

The kinematic chains are specified by a list of one degree of freedom transformations, where each transformation might

be a feature coordinate, uncertainty coordinate or a constant transformation. Each kinematic chain should contain exactly six feature coordinates and can also contain an arbitrary number of uncertainty coordinates and constant transformations.

## V. ASSEMBLY SCENARIO

The assembly case is to "best fit" a semi-compliant aircraft component to multiple surfaces. The first step of the assembly sequence is to orient the rib in such way that it is known which part of the rib that will make contact when the first search operation is applied, see Fig. 2 for the initial rib orientation. By then rotating around the first contact point a new contact with the other end of the rib can be achieved. The rib should then based on the measurements during the sequencing be positioned in the middle of the available space. A more detailed version of the assembly sequence is outlined below

1) Goto start position
2) Search for contact in $x$-direction
3) Make rotational search in $\varphi$-direction (around the $f1^a$ $z$-axis)
4) Control torques to zero
5) Back out
6) Search for contact in $z$-direction
7) Rotate until contact on the other side, in $\psi$-direction (around the $f1^a$ $x$-axis)
8) Back down
9) Search contact in negative $y$-direction
10) Search contact in positive $y$-direction
11) Calculate final position and go there

Each of these steps has a set of constraints used in the constraint-based task specification framework; the first step is made as a start-up procedure outside of the framework though. The second step is constrained to move in the $x^a$-direction and to keep the position in $y^a$, $z^a$, $\varphi^a$, $\theta^a$ and $\psi^a$ constant. The third step has a force constraint in the $x^b$ direction and position constraints in $y^b$, $z^b$, $\theta^a$ and $\psi^a$. The rotation is achieved by a velocity constraint on $\varphi^a$. Notice how the constrained outputs are changed ($x^a$ to $x^b$, $y^a$ to $y^b$ and $z^a$ to $z^b$) to allow for rotation around the contact point instead of around the center of the rib. Similar setups of constraints are used in the following steps.

The transitions between subsequent states are triggered by force sensor readings or that the robot has reached some certain position. The force readings have been lowpass-filtered to avoid transitions caused by disturbing noise. Almost each transition contains switching of some controller. A bumpless transfer is therefore implemented by making sure that the control signal is continuous before and after the switch.

## VI. EXPERIMENTAL RESULTS

It is assumed that the rib is grasped in a predicted way and that the position of the jig is known to a precision of a few centimetres, at least good enough for the initial search to hit the desired bracket on the jig. The search speed is set to around 10 mm/s for linear movements and about 1 degree/s for the rotational counterparts. The force triggers used for

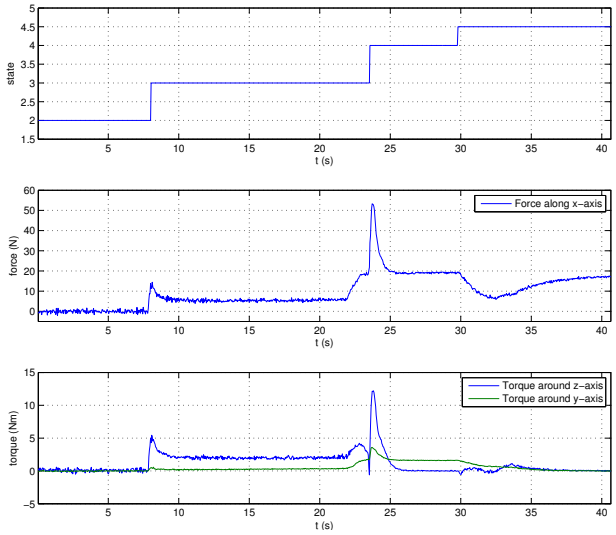


Fig. 4: State sequence, measured force, and torque during the first phases of the assembly.

transitions were set to around 10 N. The sampling time used in the control program is 4 ms, whereas the logging is made with a sampling period of 40 ms. The numbering of the states in the following figures is somewhat different from the list in section V, depending on some implementation issues.

Experimental data from the beginning of the assembly sequence can be seen in Fig. 4. The state numbered 2 is the search in the $x$-direction. A transition to the $3^{rd}$ state is triggered by a large force in the $x$-direction, which occurs at around $t = 8$ s. The $x$-direction is now force controlled and the rotational search in $\varphi^a$ begins (around the $f1^a$ $z$-axis). The next transition condition is that the torque corresponding to $\varphi^a$ should be large and negative. This happens at around $t = 23$ s. State number 4 tries to control the $\varphi^a$-torque to zero, and the next transition occurs when the torque is close enough to zero. In the last state shown in the figure (state 4.5) force control is enabled also to $\theta^a$ (corresponding to rotation around the $y$-axis).

Data from the middle part of the assembly is found in Fig. 5. State number 6 is a search in the positive $z^a$-direction and the transition condition to the next state is that the force in the $z$-direction should be large and negative. Then a rotation around the contact point should be performed to establish a contact on the other side as well. This rotation is however achieved as a concurrent force and torque control; the reference for the force is set to -15 N and the torque is simultaneously controlled to zero. The result is the desired rotation, and the transition to state number 7.5 is made when a large negative torque is detected. The force control is now made less aggressive by changing the corresponding mass and damping parameters, but some oscillations is induced anyway. The reason for this is that the contact made is especially stiff, solid metal against solid metal, see Fig. 6. When a lowpass-filtered version of the torque reaches zero the next transition is triggered, which is a move in the negative $z^a$-direction, to release the contact.

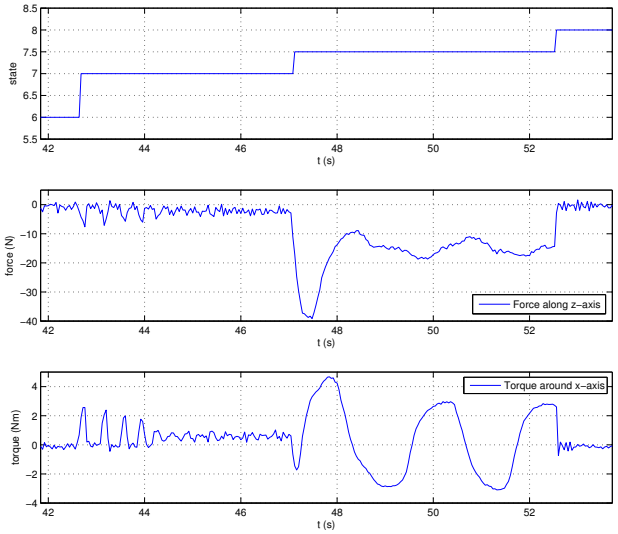


Fig. 5: State sequence, measured force and torque in the middle part of the assembly.

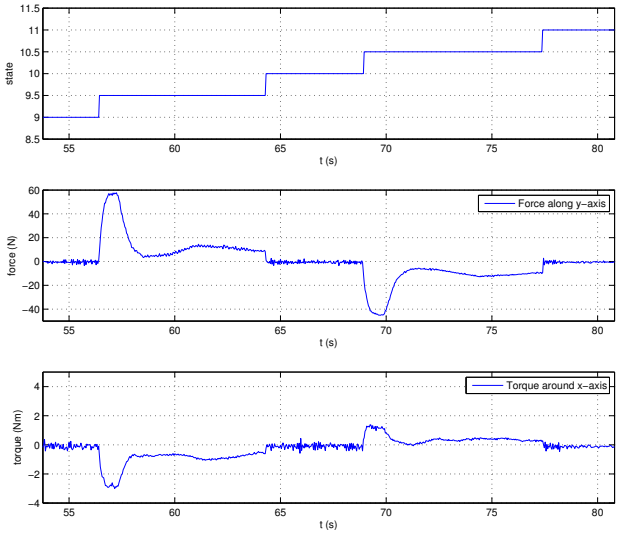


Fig. 7: State sequence, measured force and torque in the end of the assembly.

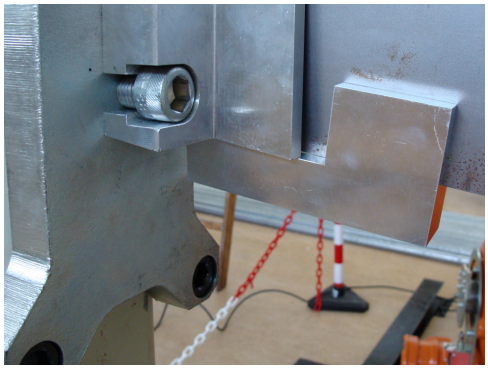

Fig. 6: Stiff contact in the vertical direction.

The last part of the assembly is illustrated in Fig. 7, which is the alignment in the $y^a$-direction. State number 9 is a search in negative $y^a$, when contact is made the contact force is controlled to 10 N. A transition to the next state is made when the force is stabilized. The same procedure is now performed in the positive $y^a$-direction. The large contact forces seen in the figure is also an effect of stiff contacts, which also can be anticipated by studying the contact that will be made in the horizontal direction in Fig. 6.

Some limitations on the assembly speed can be seen in Figs. 4, 5 and 7. The first one is the transients originating from when contact is made; they are especially large when contact is made after the rotational search at $t = 23$ s in Fig. 4 and at $t = 57$ s in Fig. 7. The size of the transients grows fast with increasing search speeds, and the speed has to be kept quite low to avoid damaging the equipment. Another problem occurs when one wants to apply force control simultaneously in several directions. If the control parameters are just a little too aggressive the result will be poorly damped or giving rise to unwanted limit cycles. Symptoms of the problem are shown after the transition from state 7 to 7.5, where obvious oscillations in the $\psi^a$-torque can be seen (torque around $x$-axis in Fig. 5). A consequence of using a well-damped impedance controller is that it takes quite some time for it to reach the desired force, which clearly can be seen in the force plot in state number 4.5 (Fig. 4).

Velocity data from the whole performed experiment can be found in Fig. 8. The uppermost plot shows the current state, the middle the linear velocities for loop $a$ and the lowermost plot the rotational velocities in loop $a$. The measured velocity is given when the corresponding feature coordinate is position/velocity controlled, but when the coordinate is force controlled the control signal is given (the desired velocity). The initial search, state number 2, is characterised by a velocity in the negative $x$-direction. The next step is the rotational search, which here can be seen by a non-zero velocity for the $\varphi^a$ coordinate. There is also a velocity in the $x^a$ coordinate, as the rotation is made with respect to loop $b$. The alignment steps following are harder to recognize, state 4 and 4.5. Then there is the search phase in $z^a$ and the alignment in $\psi^a$. The last part is search for contact in first negative $y^a$ and then positive $y^a$ and finally position control to the calculated midpoint.

The contact torque arising when the rotational search makes contact at $t$=23 s gives rise to a rapid response for the force controller for $\varphi^a$, which is clearly seen in the rotational velocity plot. Another thing to notice is the oscillation in the $\psi^a$ velocity around $t$=50 s, which previously also was seen in the torque plot in Fig. 5.

## VII. DISCUSSION

The fully automated assembly sequence works fine, although it is rather slow. This is a problem that arises from relying solely on force feedback. The search operations have to be made quite slow, as the environment is rather stiff and a fast impact will then cause big force transients. Another problem with a stiff environment is that it requires fast

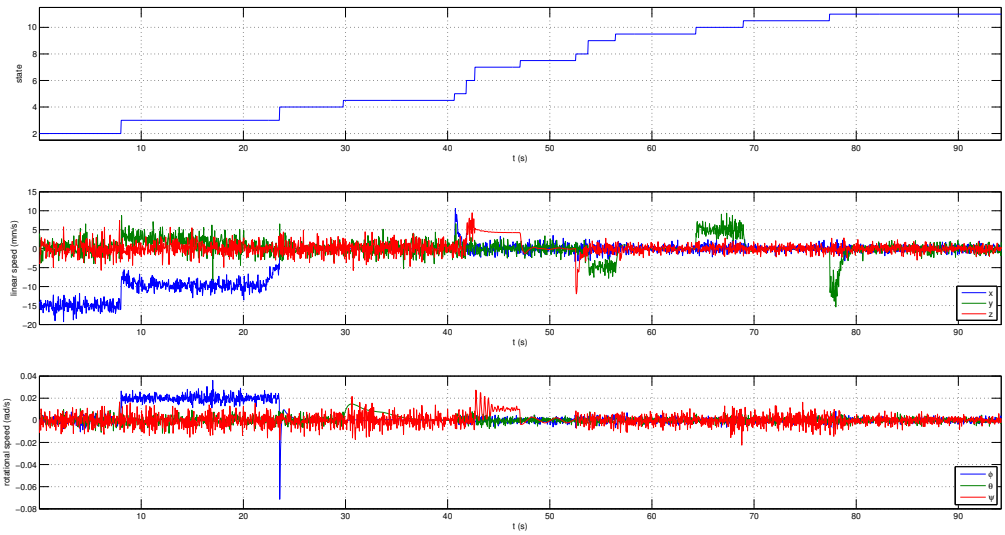


Fig. 8: Velocity data from the whole assembly sequence. The uppermost diagram shows the state sequence, the middle the linear velocities and the lowermost the rotational velocities.

feedback, as small motions cause large forces that have to be reacted upon. There may also be a problem with time delays caused by sampling, network connections and calculations in the controller. This can cause stability problems when choosing too aggressive control parameters. Therefore it would be desirable to use a more extensive feed-forward part in the control once the first assembly has been performed. Another way to speed up the sequence would be to use a vision system that can tell when a contact is close, and telling the controller when to start the slow search. The assembly could also be made in a semi-automatic fashion. An operator could then make the initial positioning with the robot in a lead-through mode (i.e., using the force control functionality to allow the operator to manually move ('lead') the robot manipulator).

A shortcoming with the current software implementation is that it is relying on the user to choose a singularity-free representation for the kinematic chains. The Euler angles chosen in the experimental task has a singularity that can cause troubles in some parts of the robot workspace. This problem is something that the software has to be able to take care of.

All control parameters have to be manually tuned in the software today, and a future step is to add a self-tuning mechanism. With this feature it would be possible to start up with slow and well-damped controllers that are tuned to be faster during runtime. This would simplify the task specification, as there would be fewer choices to make for the user.

The current implementation of the assembly sequence utilises torque measurements. Another way to do it would be to control the contact force in two places, i.e. the two contact points. It is possible to calculate both forces if it is assumed that the contact points are known, but the method will fail if a contact is made in an unexpected position. It would simplify the task specification using this second approach, it is, however, preferable to use the first approach from a robustness point of view.

## VIII. CONCLUSIONS

This paper describes how an assembly task from the aircraft industry can be modeled using the constraint-based task specification framework. A software implementation based on the above mentioned framework is described and experimental results are shown. The software is such that it is easy to reuse and it can easily be used to perform other types of force controlled assemblies.

## REFERENCES

[1] K.J. Åström and B. Wittenmark. *Computer-controlled systems: theory and design*. Prentice Hall New York, 1996.

[2] A. Blomdell, G. Bolmsjö, T. Brogårdh, P. Cederberg, M. Isaksson, R. Johansson, M. Haage, K. Nilsson, M. Olsson, T. Olsson, A. Robertsson, and J. Wang. Extending an industrial robot controller–Implementation and applications of a fast open sensor interface. *IEEE Robotics & Automation Magazine*, 12(3):85–94, September 2005.

[3] A. Blomdell, I. Dressler, K. Nilsson, and A. Robertsson. Flexible application development and high-performance motion control based on external sensing and reconfiguration of abb industrial robot controllers. In *Proc. ICRA 2010 Workshop on Innovative Robot Control Architectures for Demanding (Research) Applications*, Anchorage, Alaska, USA, June 2010.

[4] H. Bruyninckx. Open robot control software. http://www.orocos.org/. Last visited 2010.

[5] J. De Schutter, T. De Laet, J. Rutgeerts, W. Decré, R. Smits, E. Aertbeliën, K. Claes, and H. Bruyninckx. Constraint-based task specification and estimation for sensor-based robot systems in the presence of geometric uncertainty. *The International Journal of Robotics Research*, 26(5):433, 2007.

[6] M. Jonsson, T. Murray, A. Robertsson, A. Stolt, G. Ossbahr, and K. Nilsson. Force Feedback for Assembly of Aircraft Structures. In *Proc. 2010 SAE Aerospace Manufacturing and Automated Fastening Conference*, Wichita, KS, USA, September 2010.

[7] T. Olsson, M. Haage, H. Kihlman, R. Johansson, K. Nilsson, A. Robertsson, M. Björkman, R. Isaksson, G. Ossbahr, and T. Brogårdh. Cost-efficient drilling using industrial robots with high-bandwidth force feedback. *Robotics and Computer-Integrated Manufacturing*, 26:24–38, January 2010.

[8] R. Smits. *Robot Skills: Design of a Constraint-Based Methodology and Software Support*. PhD thesis, K.U. Leuven, Belgium, 2010.