Deutsches Herzzentrum München
Klinik für Herz- und Gefäßchirurgie
Prof. Dr. med. Robert Bauernschmitt

Technical University Munich
Robotics & Embedded Systems
Prof. Dr.-Ing. Alois Knoll

Mensch-Maschine Skilltransfer in der robotergestützten Chirurgie

H. Mayer*°, I. Nagy* und A. Knoll*
E.U. Braun° und R. Bauernschmitt°

*Informatik VI – Robotik und Echtzeitsysteme
°Deutsches Herzzentrum München

Technische Universität München

SFB 453
Wirklichkeitsnahe Telepräsenz und Teleaktion

Übersicht

- Einführung
- Erstes System (EndoPAR) und Evaluation
- Zweites System (ARAMIS) am Deutschen Herzzentrum München
- Anwendungen
- Programmierter Knoten
- Skilltransfer
- Zusammenfassung

Motivation

- Medizinrobotik hat das Potential einen ähnlichen Stellenwert für chirurgische Verfahren zu erlangen, wie dies vor ca. 30 Jahren bei Industrierobotern für die industrielle Fertigung der Fall war
  - Skalierung,
  - Zugriffsmöglichkeiten,
  - Telepräsenz,
  - Handhabung und Geschwindigkeit
- Dieses Potential lässt sich nur dann nutzen, wenn Roboter die folgenden Eigenschaften erlangen
  - Höchste Flexibilität und die Fähigkeit sich automatisch an vorher unbekannte Aufgaben und Umgebungen anzupassen
  - Automatische Erkennung chirurgischer Handlungsabläufe und die Fähigkeit damit eigene Steuerungsprogramme zu verbessern (Lernen durch Nachahmen)

  mit anderen Worten: Roboter müssen intelligenter werden um so den Chirurgen von untergeordneten Tätigkeiten zu entlasten

Kooperationspartner

Projekt wird von der Deutsche Forschungsgemeinschaft finanziert
Sonderforschungsbereich 453 ("Telepräsenz und Teleaktion")

Partner 1: **Deutsches Herzzentrum München**	Partner 2: **Institut für Informatik**
Umfasst 3 klinische Abteilungen und 3 Forschungsinstitute; 700 Mitarbeiter und Betten für bis zu 170 Patienten	Arbeitsgruppen in Medizininformatik, Robotik und Echtzeitsysteme und Virtual/Augmented Reality
	

Momentaner Stand der Technik in der Herzchirurgie

- Telemanipulatoren für MI-Herzchirurgie sind bereits kommerziell verfügbar
- “Chirurg in der Regelungsschleife" hat stets die vollständige Kontrolle
- Keine Kraftrückkopplung, nur visuelles Feedback → sensorische Substitution erforderlich

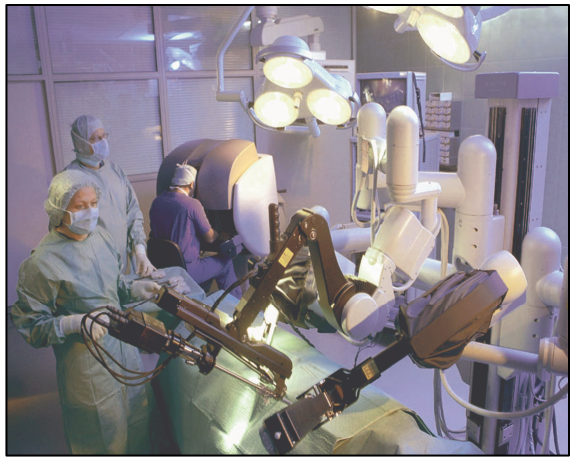
→ Aus der Sicht der Chirurgen sind komplexe Aufgaben (z.B. Knoten) nur umständlich zu bewältigen:

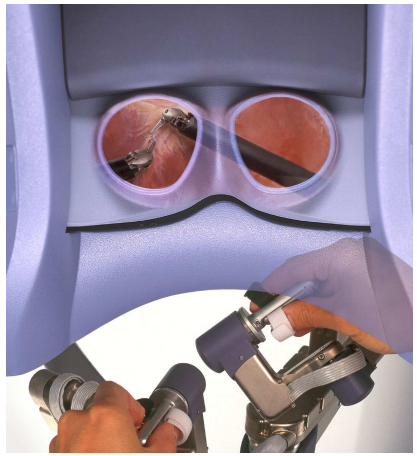
  - Fehleranfällig (häufiger Fadenriss)
  - Sehr zeitaufwändig
  - Ermüdung

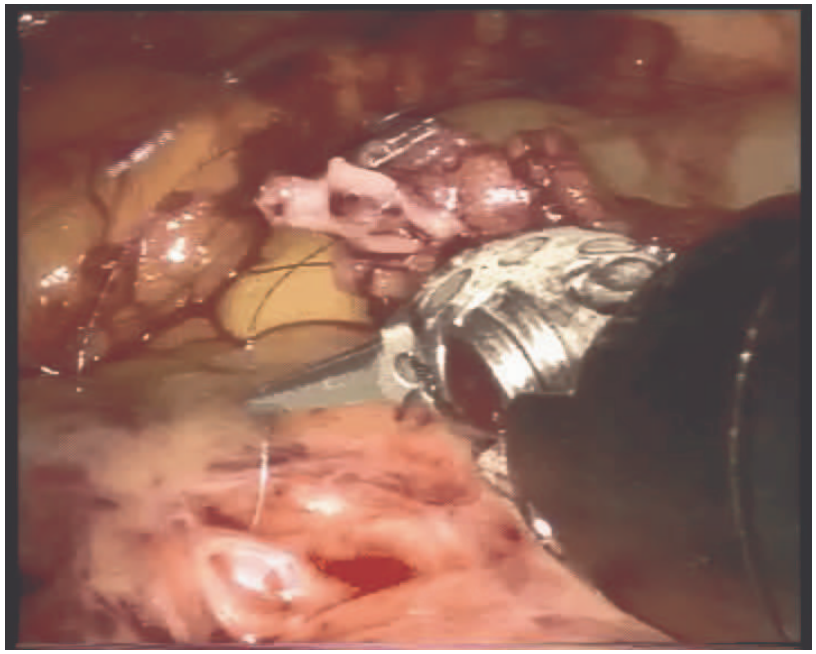
- Navigation nur optisch möglich (& präoperative Angiographien):

  - Stereosicht während der OP mit einem möglichst großen Sichtfeld
  - on-line Projektion von Patientendaten

  wären wünschenswert

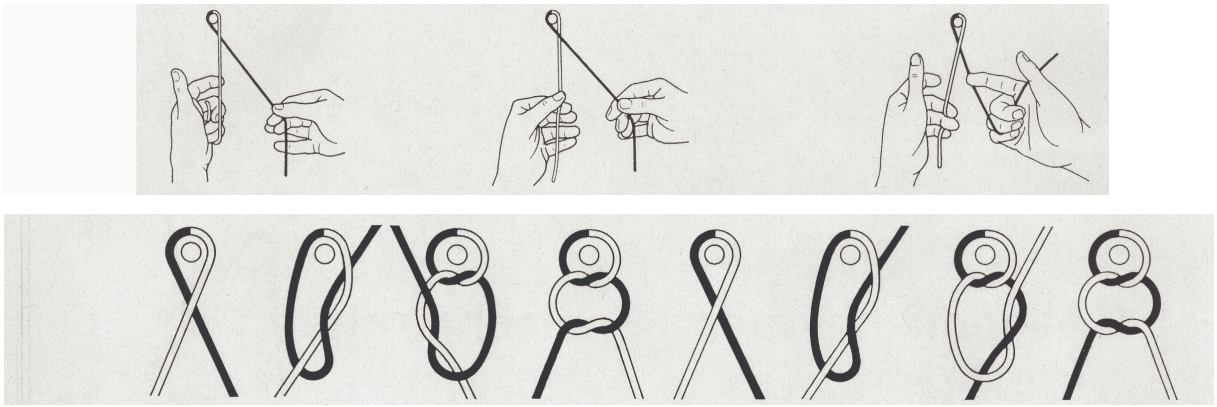






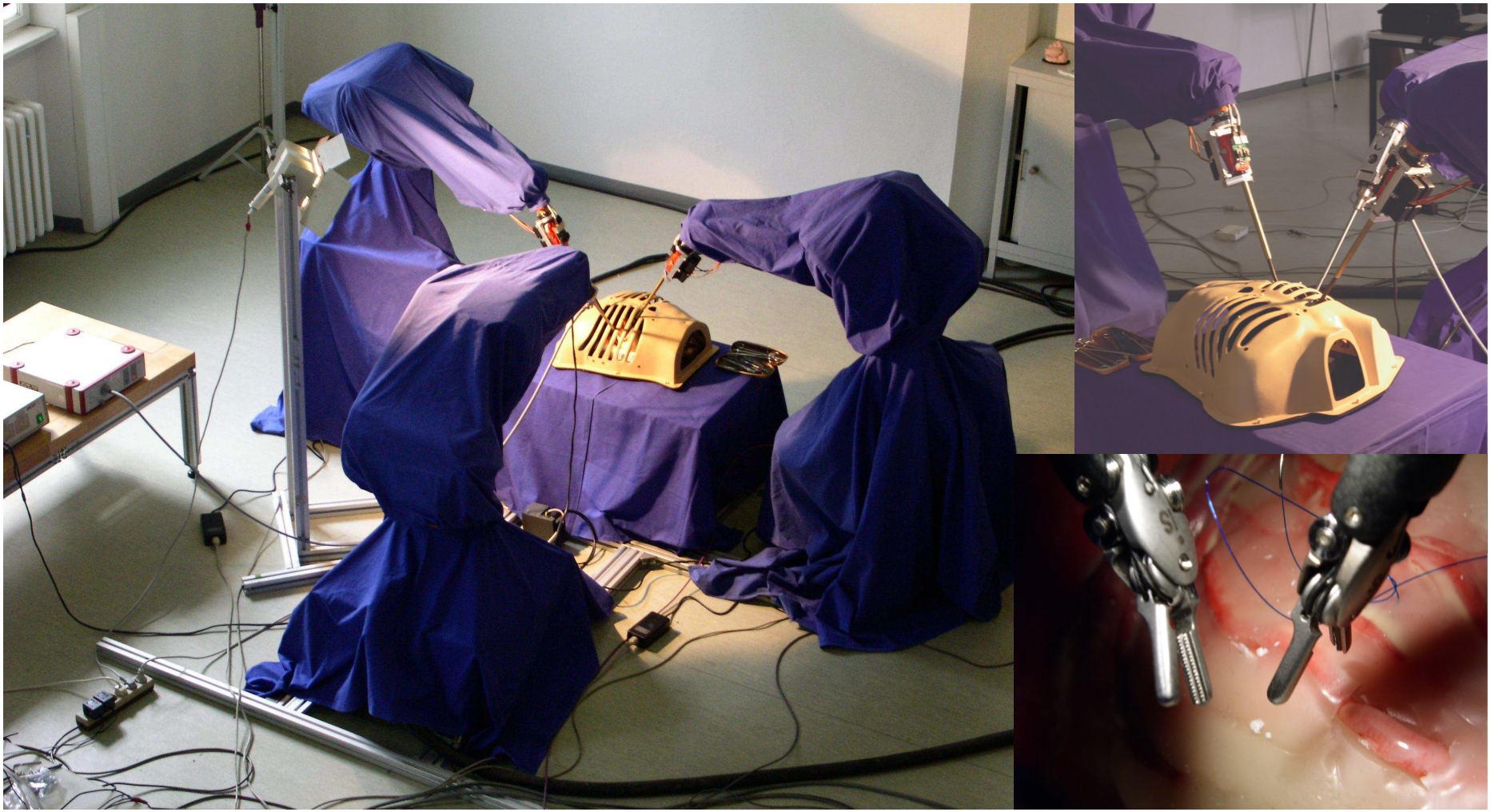
Entwicklungskriterien ...

- Voll funktionsfähiges System (nicht für den Einsatz im Operationssaal aber für Trainingszwecke geeignet)
- Einsatz von Standardkomponenten (Roboter, Sensorik, Software)
  Vollständig dokumentierte, offene Programmierschnittstelle
- Alle System-Parameters werden aufgezeichnet und sind zentral verfügbar (Roboter vs. Telemanipulator) – notwendig für Teilautonomie
- Hochauflösende Kraftrückkopplung in drei Richtungen
- Alle Trajektorien und Kräfte müssen synchronisiert aufgezeichnet werden

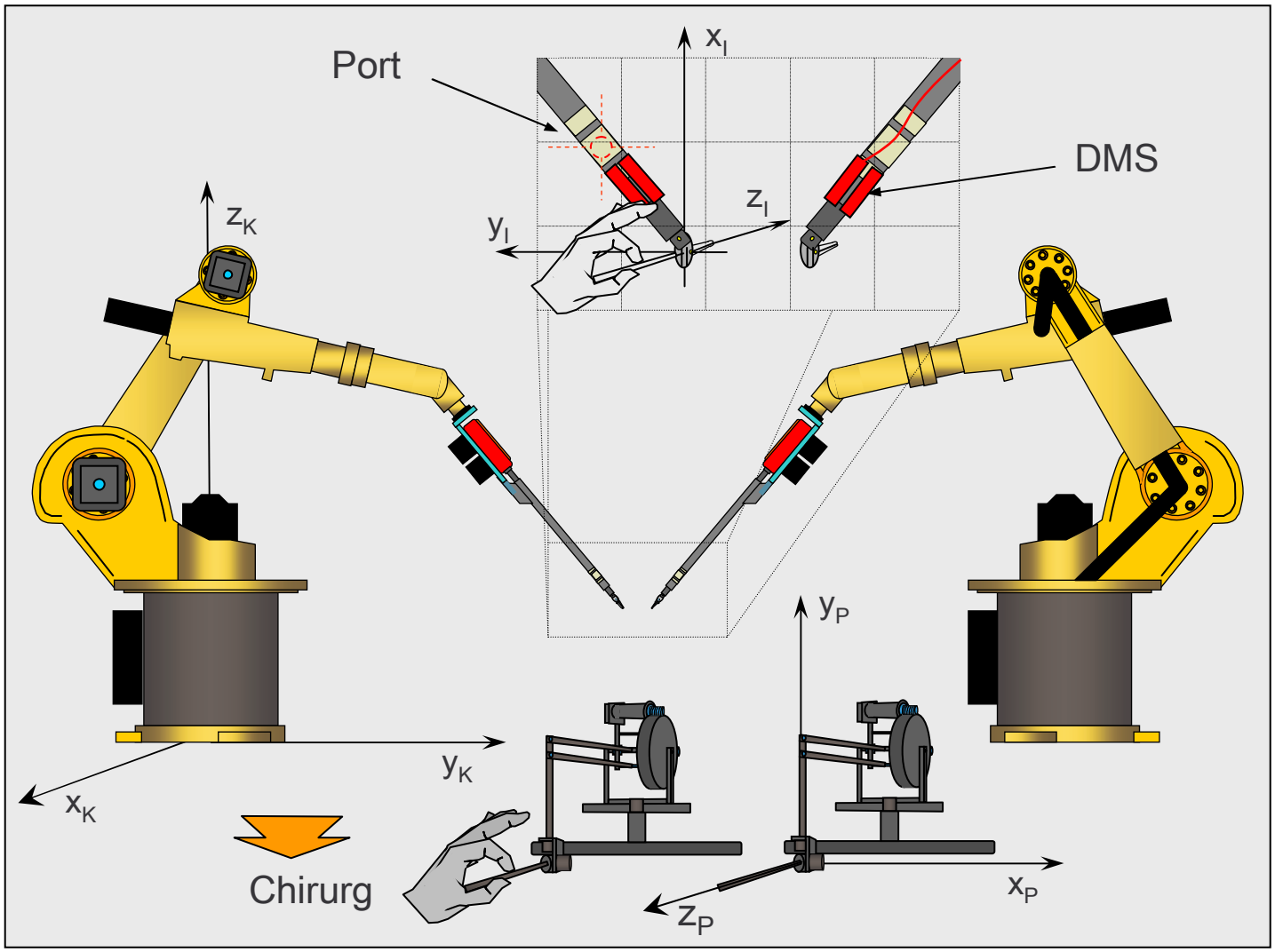


Robotersystem zur Evaluation von Teilautonomie

... und Implementierung (erste Version des Systems 2004)



Robotersystem zur Evaluation von Kraftrückkopplung



PHANToM (Position/Orientierung)

Trokar-Kinematik

Instrument / Kamera (Position/Orientierung)

Kontakt-Information

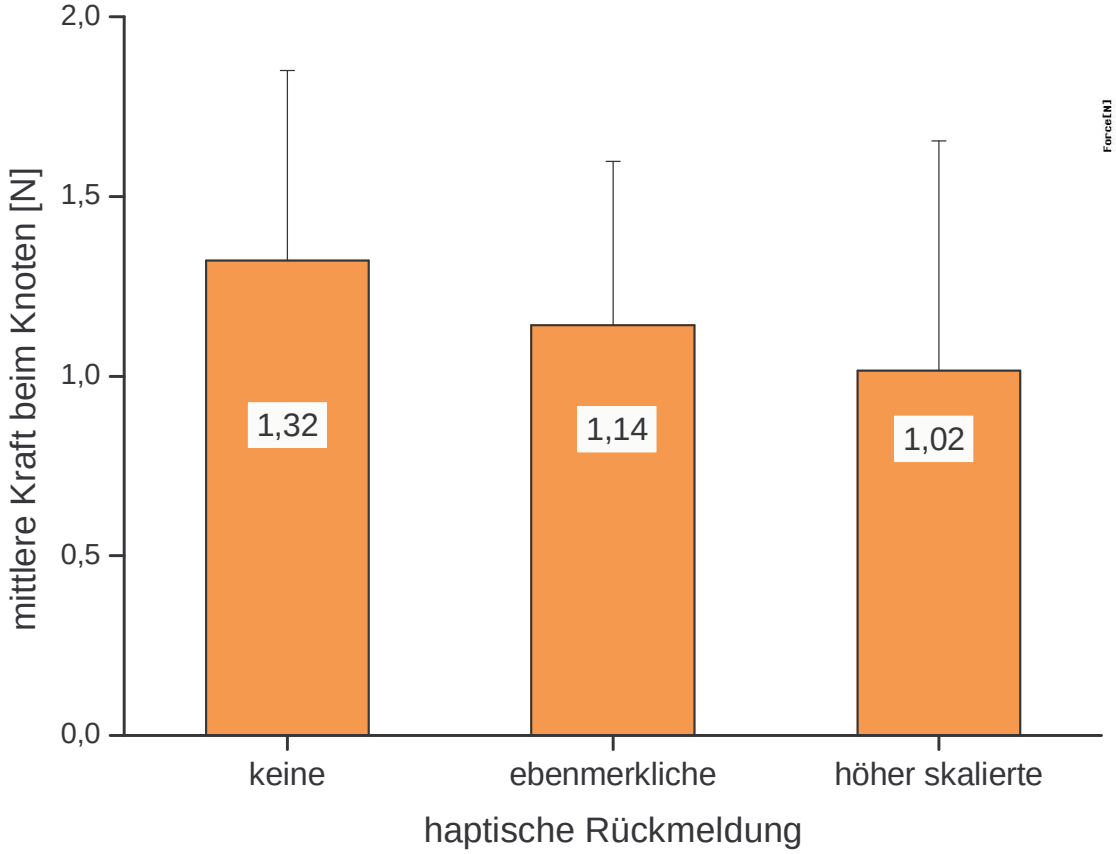
Dehnmessstreifen (x-y-z Kräfte)

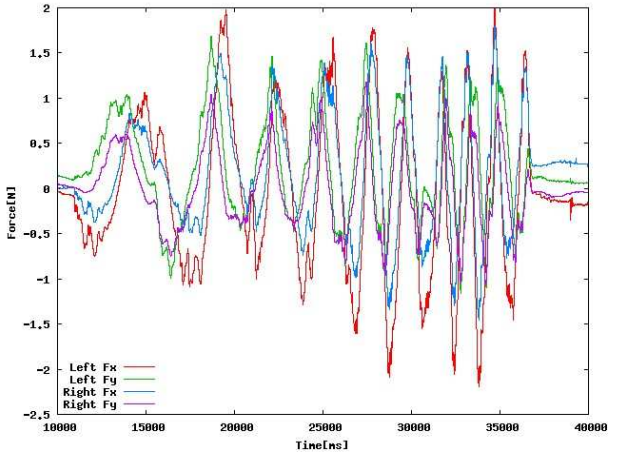
Transformation

PHANToM (Kraftrückkopplung)

Evaluation des Systems durch Chirurgen

Auftretende Kräfte

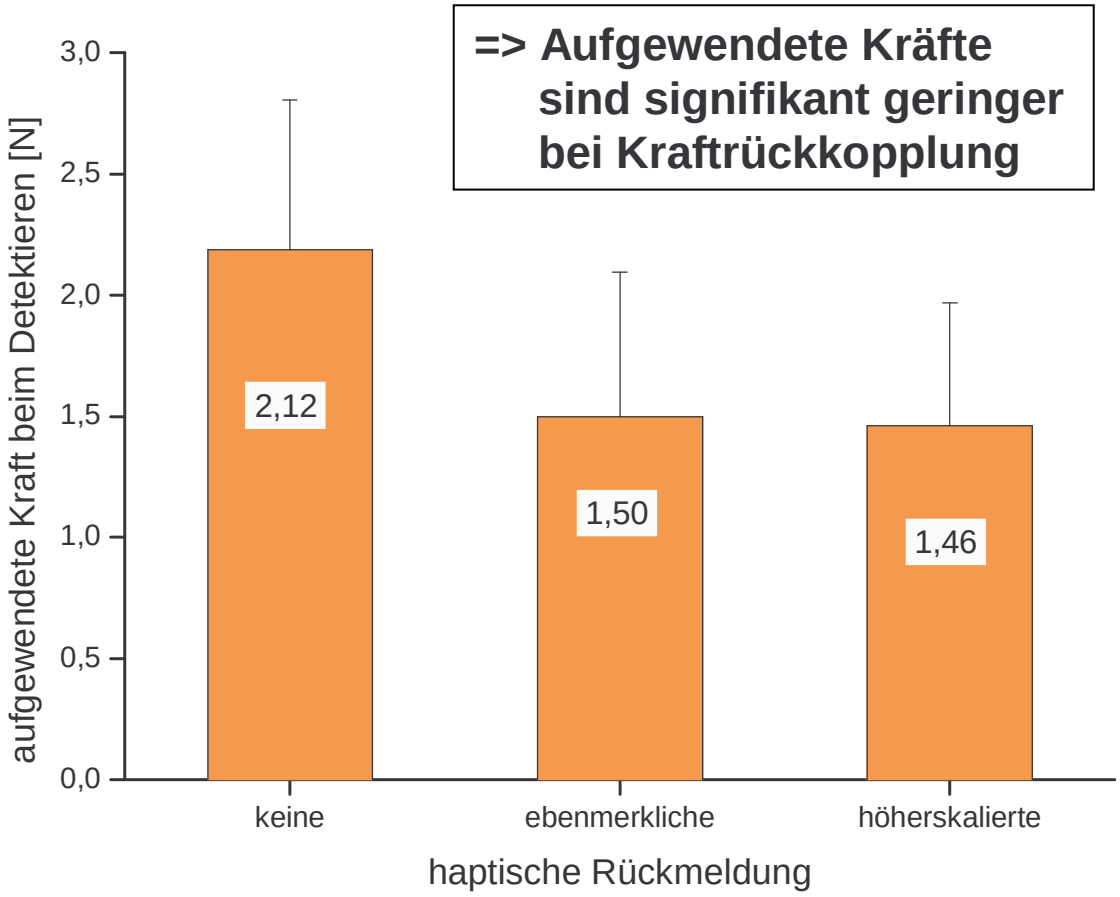


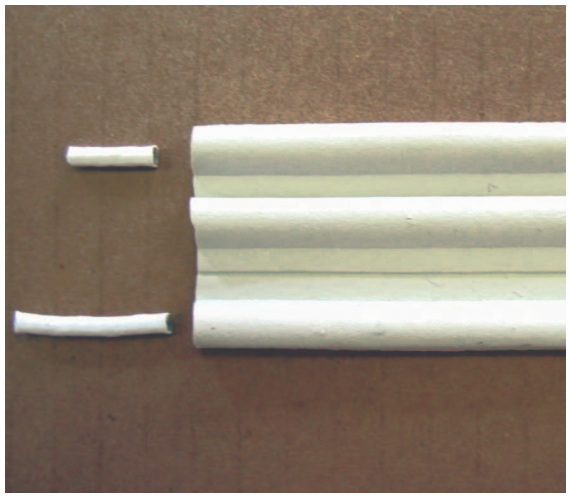


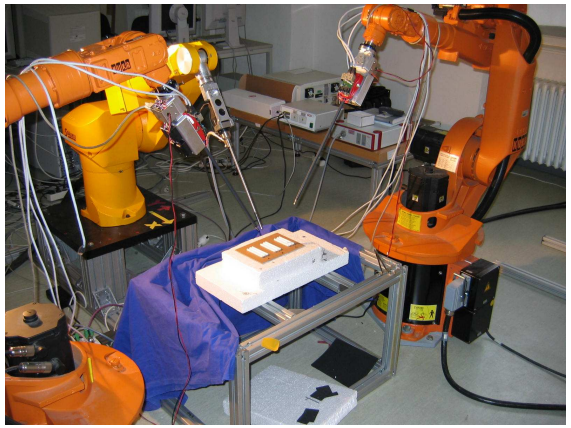
=> Aufgewendete Kräfte mit Kraftrückkopplung signifikant geringer

Evaluation des Systems durch Chirurgen

Auftretende Kräfte bei der Detektion

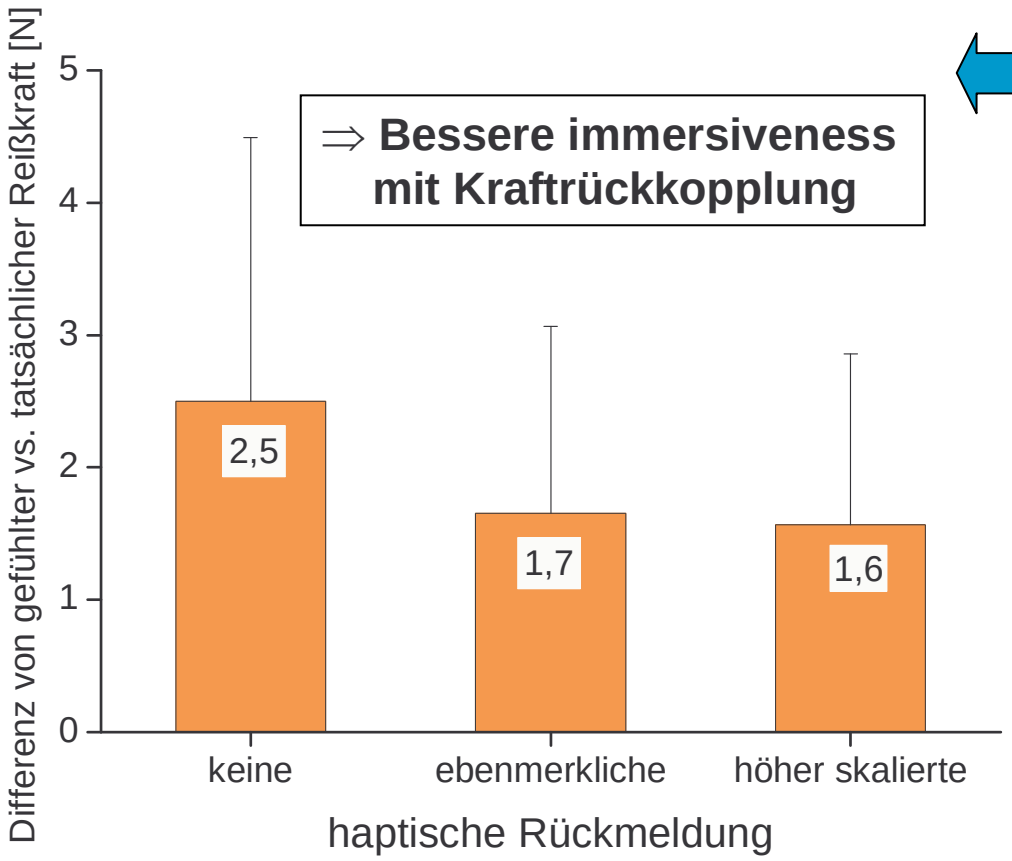




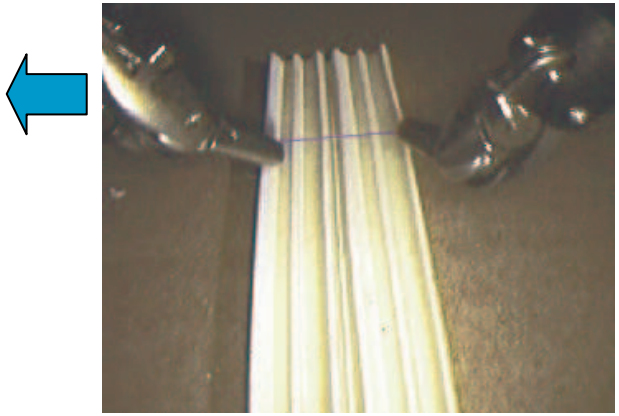


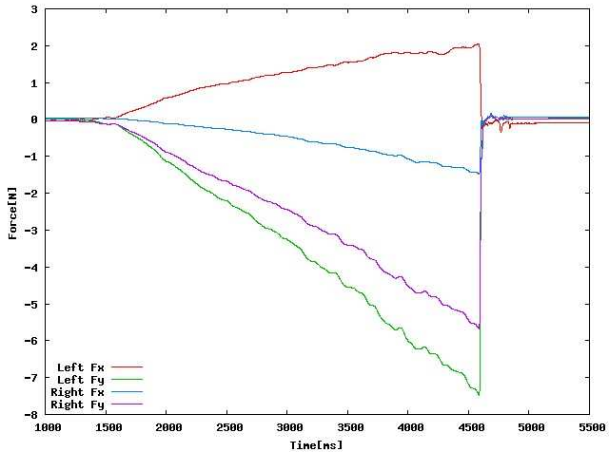
Evaluation des Systems durch Chirurgen

„Immersivenes“: Fadenriss

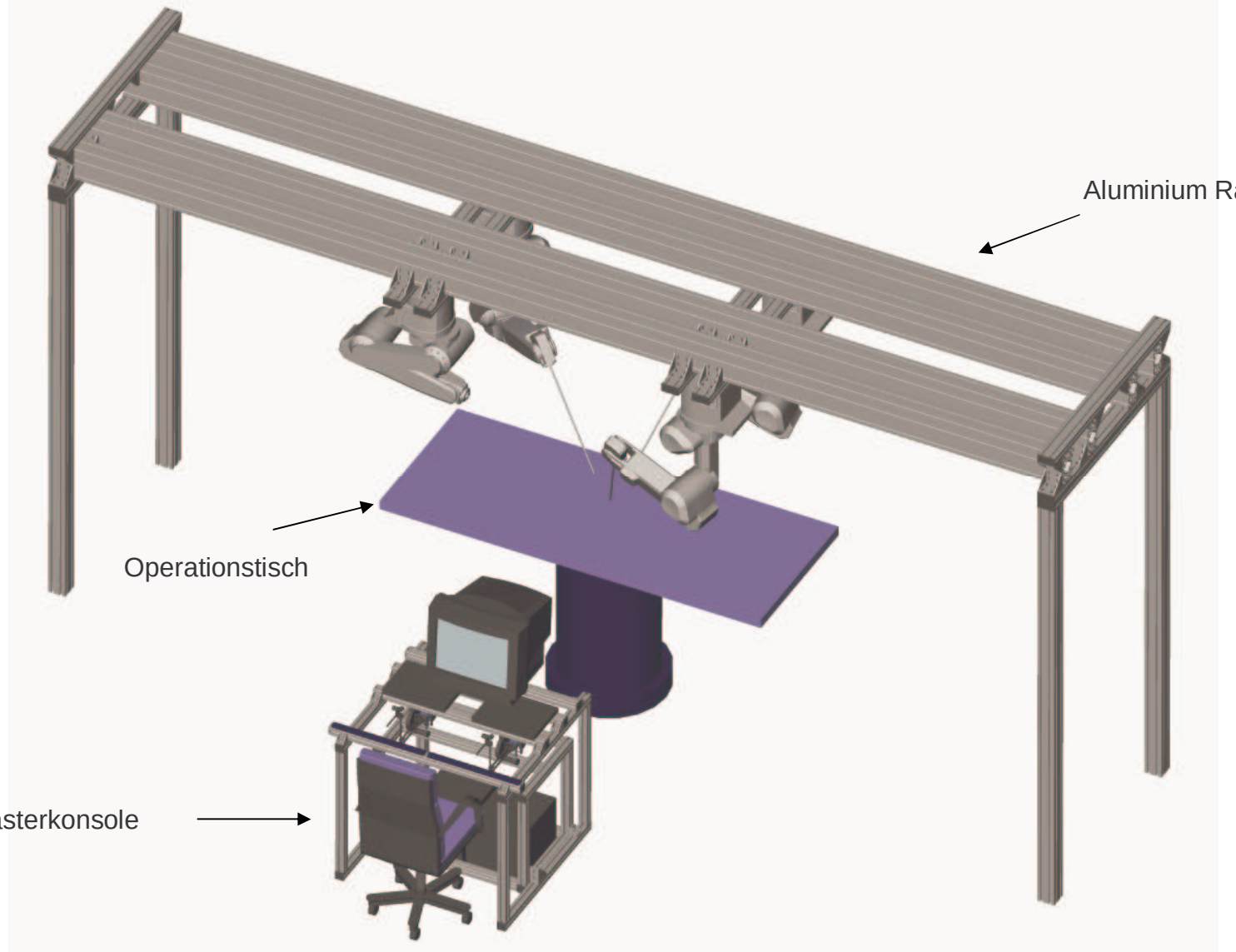


⇒ **Bessere immersiveness mit Kraftrückkopplung**





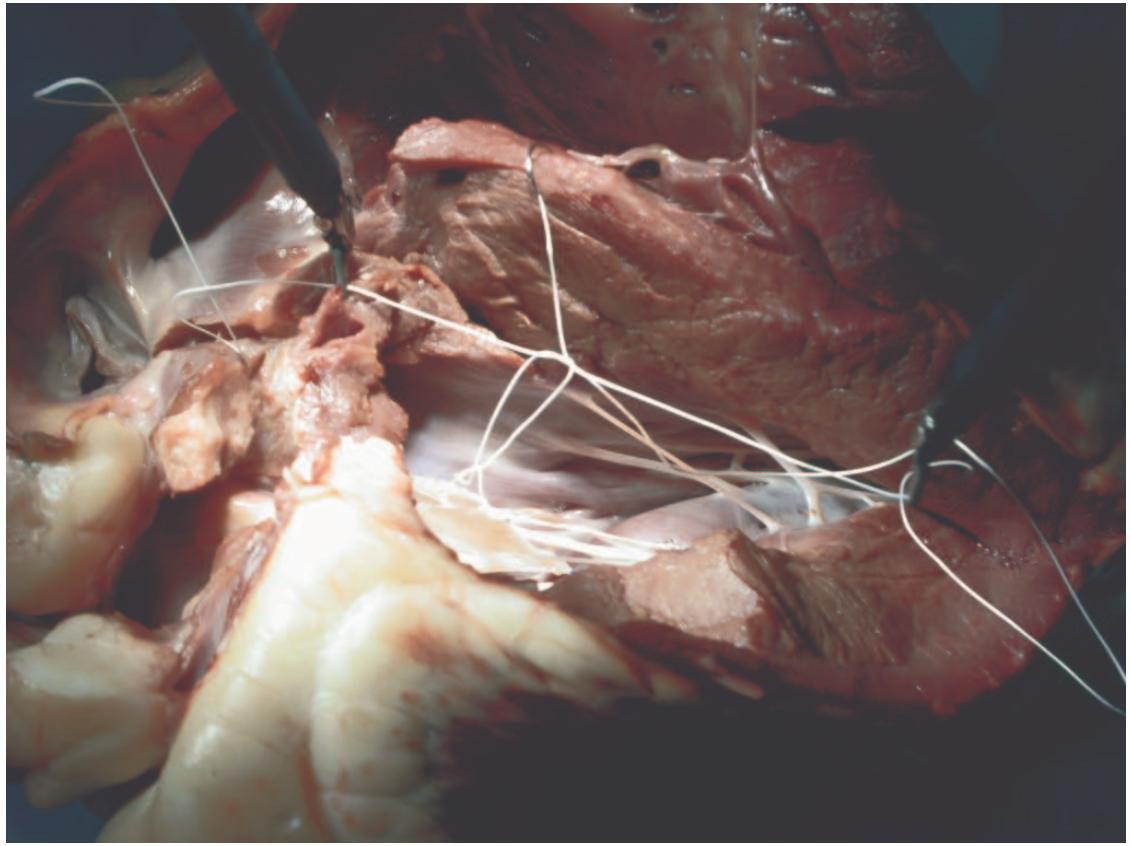
Neuer Aufbau am Deutschen Herzzentrum München

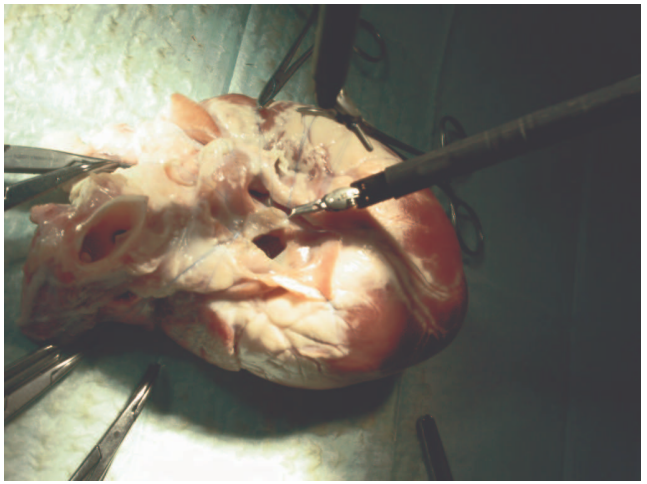


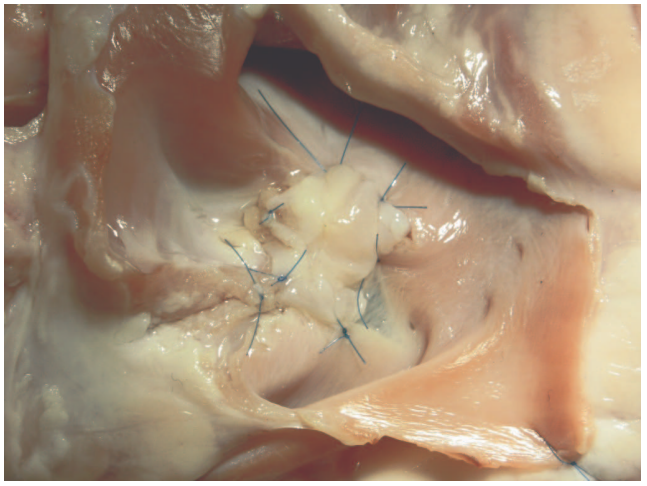
Neuer Aufbau am Deutschen Herzzentrum München

- Drei Roboterarme mit Instrumenten + ein Kameraroboter
- 2 DoF Kraftmessung (SGS-Technologie) + Feedback
- Masterkonsole mit 3D - Display
- Betriebssystem Linux 2.6.20 64bit (low – latency kernel)

Durchgeführte Versuche



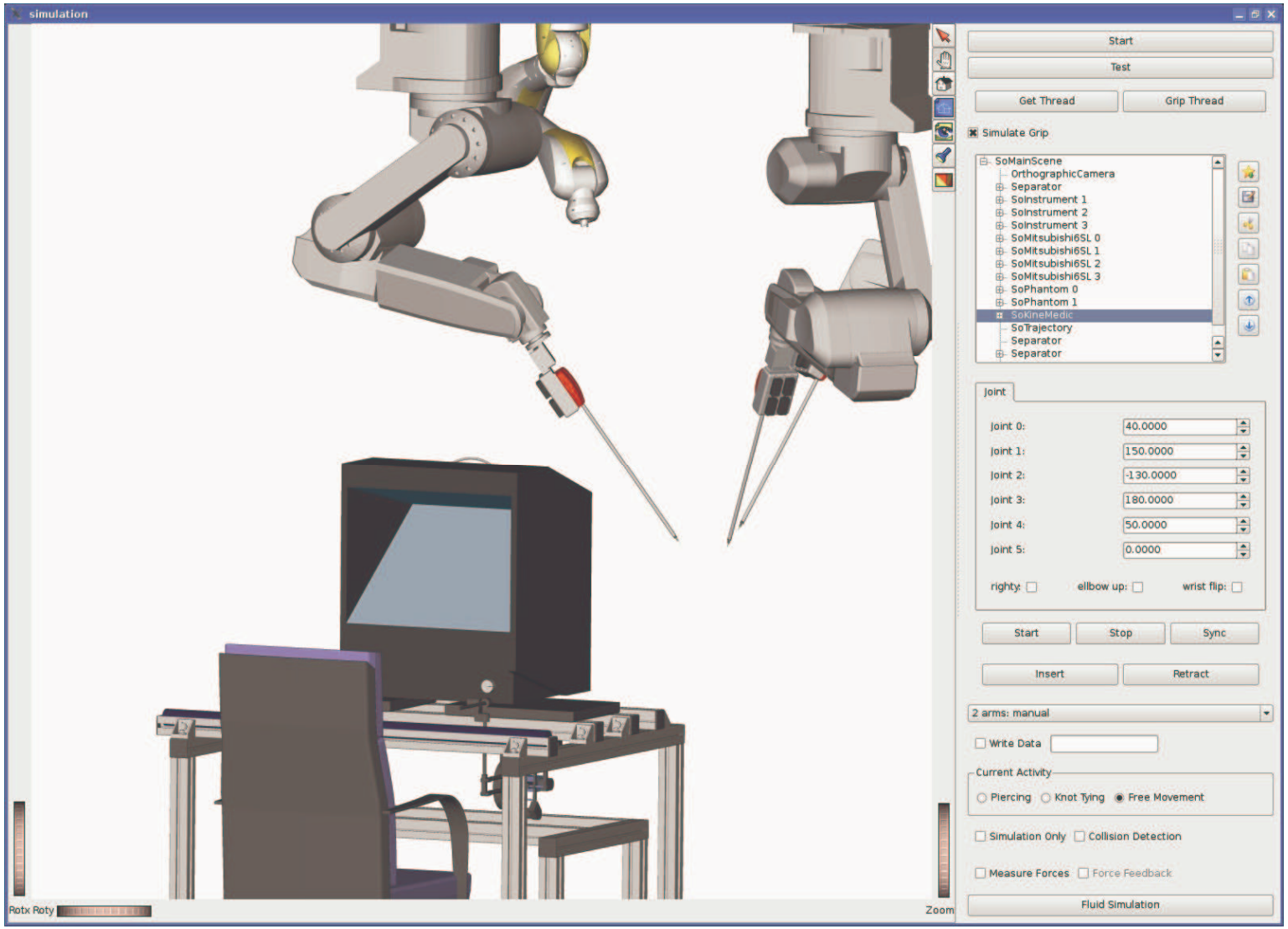




Ersatz einer Papillarsehne durch Gore-Tex® Faden

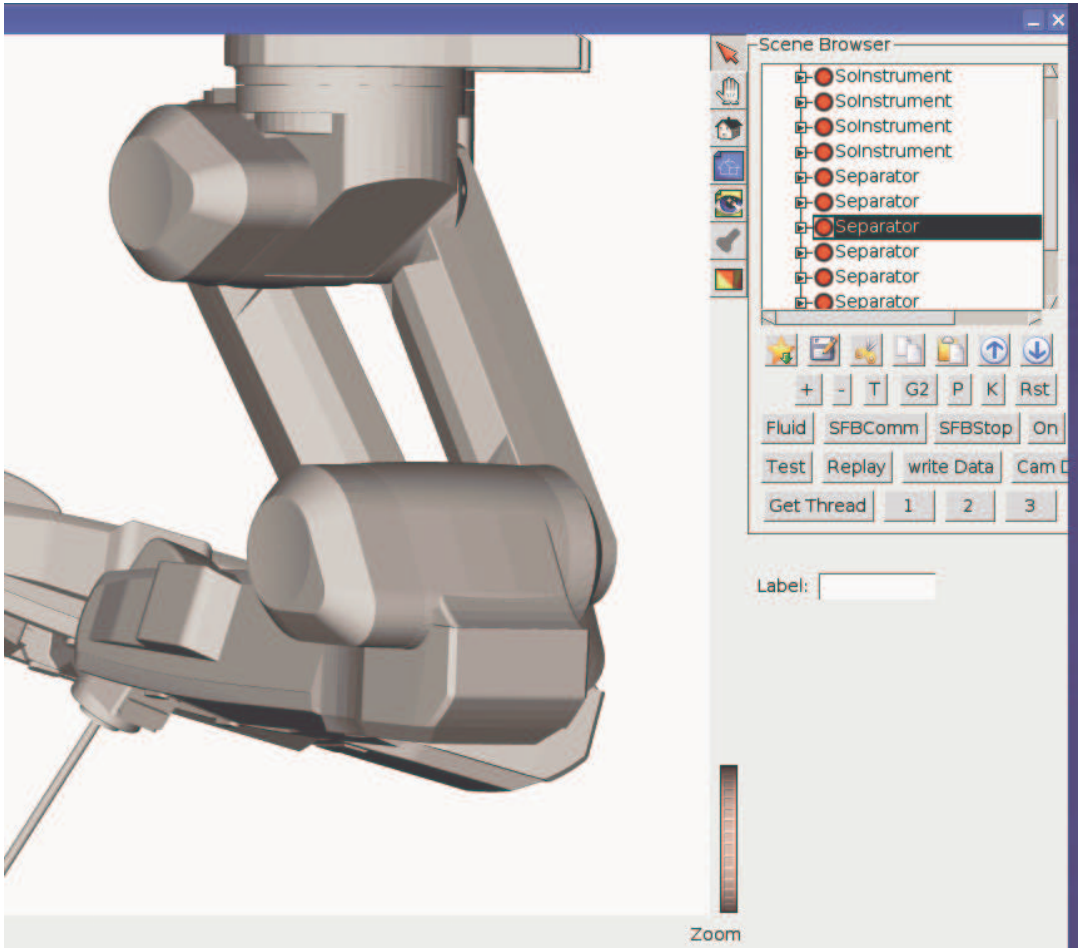
Verschluss eines Vorhofseptumdefekts (ASD) →

Grafische Benutzeroberfläche



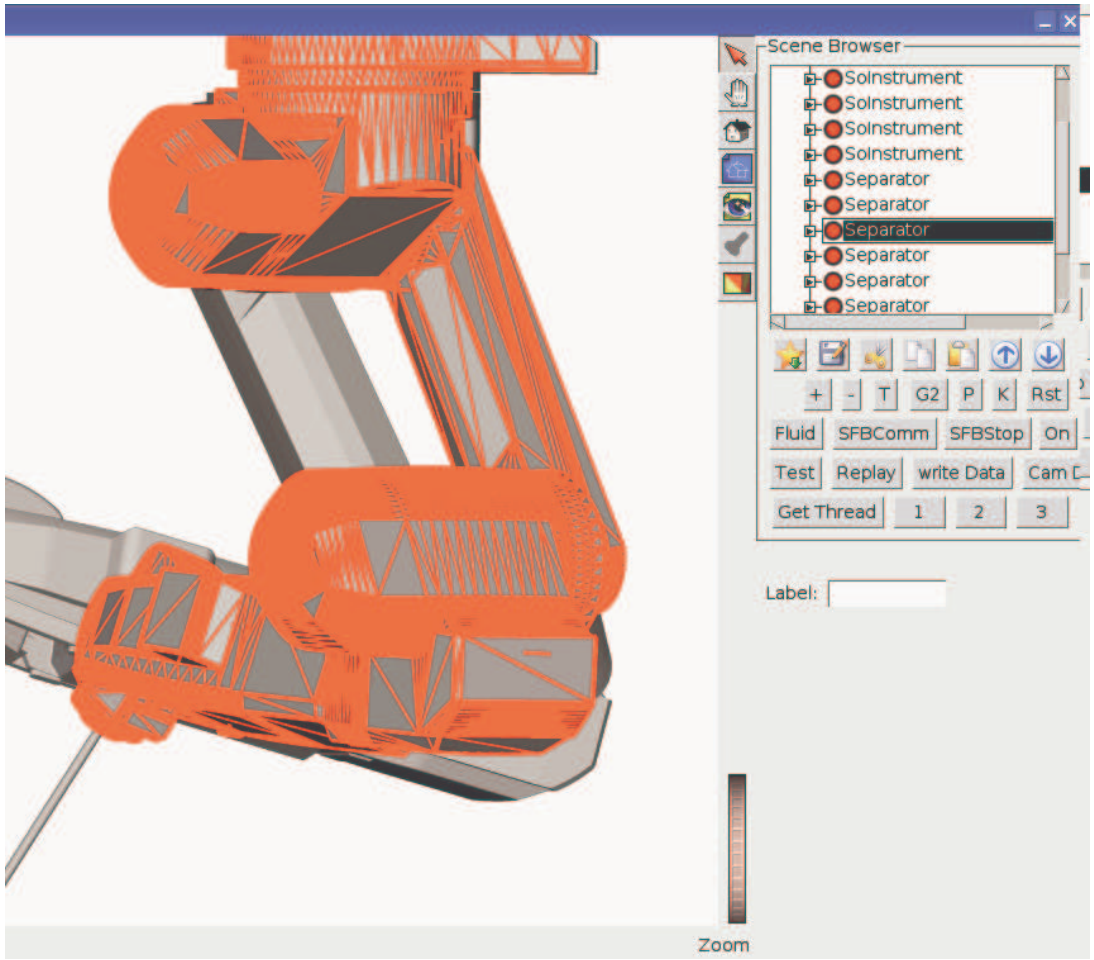
Darstellung der Szene und Kontextmenüs

- Szene wird durch eine Baumstruktur repräsentiert und in einem OpenGL-Fenster graphisch dargestellt
- Für jeden Knotentyp existiert ein Kontextmenü zur Einstellung der relevanten Parameter
- Änderungen in der Szene werden zuerst in der 3D Darstellung angezeigt und können danach in das reale System übernommen werden
- Mit dem „Scene Browser“ können Änderungen am Szenenbaum vorgenommen werden (einfache CAD-Funktionalität



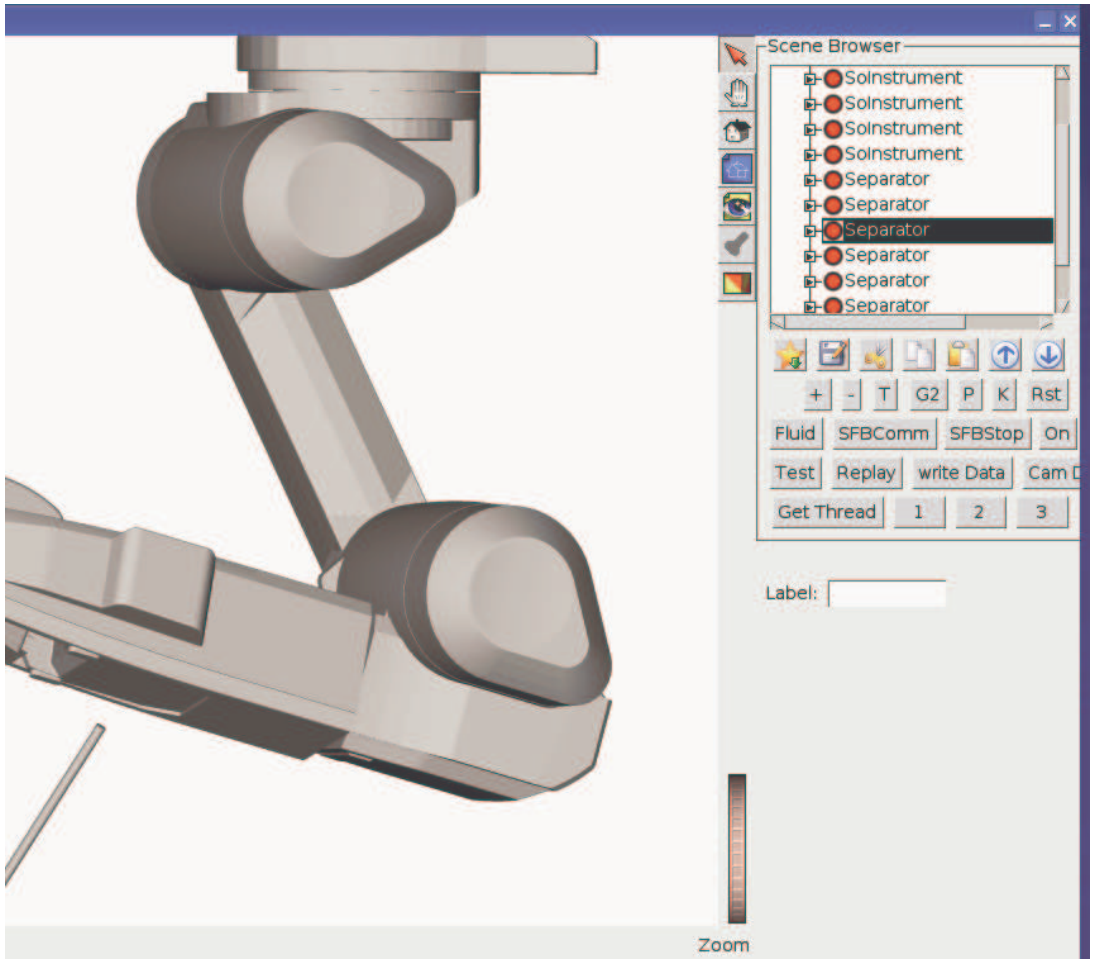
Scene Browser and Context Menus

- Szene wird durch eine Baumstruktur repräsentiert und in einem OpenGL-Fenster graphisch dargestellt
- Für jeden Knotentyp existiert ein Kontextmenü zur Einstellung der relevanten Parameter
- Änderungen in der Szene werden zuerst in der 3D Darstellung angezeigt und können danach in das reale System übernommen werden
- Mit dem „Scene Browser“ können Änderungen am Szenenbaum vorgenommen werden (einfache CAD-Funktionalität



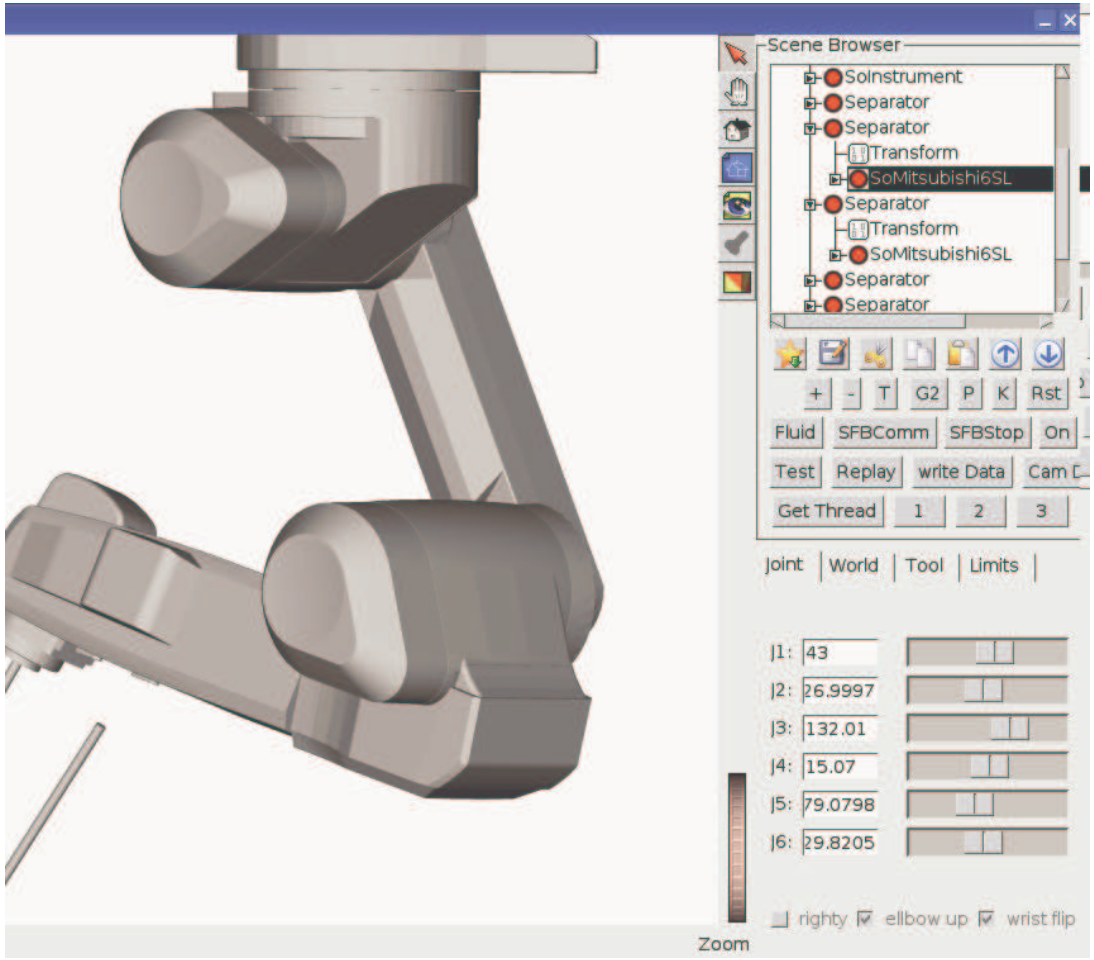
Scene Browser and Context Menus

- Szene wird durch eine Baumstruktur repräsentiert und in einem OpenGL-Fenster graphisch dargestellt
- Für jeden Knotentyp existiert ein Kontextmenü zur Einstellung der relevanten Parameter
- Änderungen in der Szene werden zuerst in der 3D Darstellung angezeigt und können danach in das reale System übernommen werden
- Mit dem „Scene Browser“ können Änderungen am Szenenbaum vorgenommen werden (einfache CAD-Funktionalität



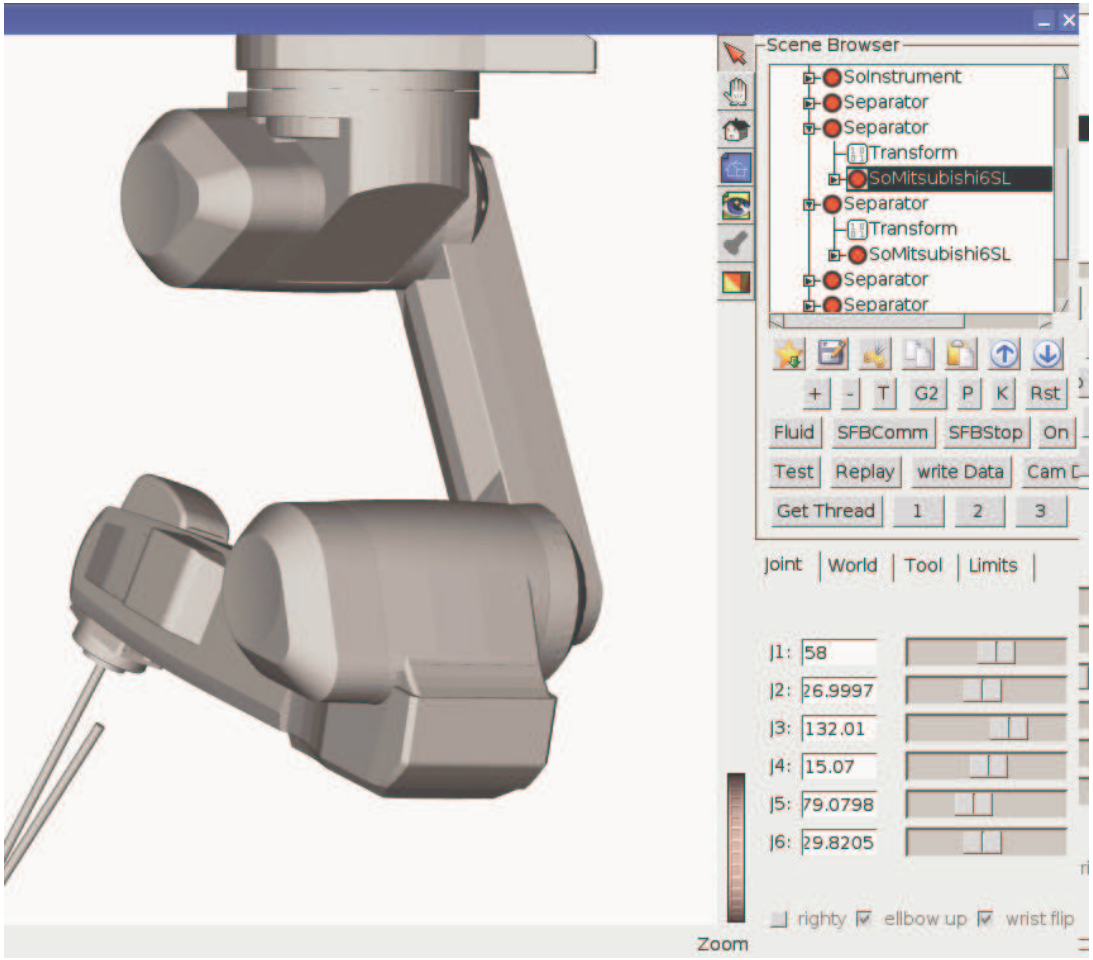
Scene Browser and Context Menus

- Szene wird durch eine Baumstruktur repräsentiert und in einem OpenGL-Fenster graphisch dargestellt
- Für jeden Knotentyp existiert ein Kontextmenü zur Einstellung der relevanten Parameter
- Änderungen in der Szene werden zuerst in der 3D Darstellung angezeigt und können danach in das reale System übernommen werden
- Mit dem „Scene Browser“ können Änderungen am Szenenbaum vorgenommen werden (einfache CAD-Funktionalität



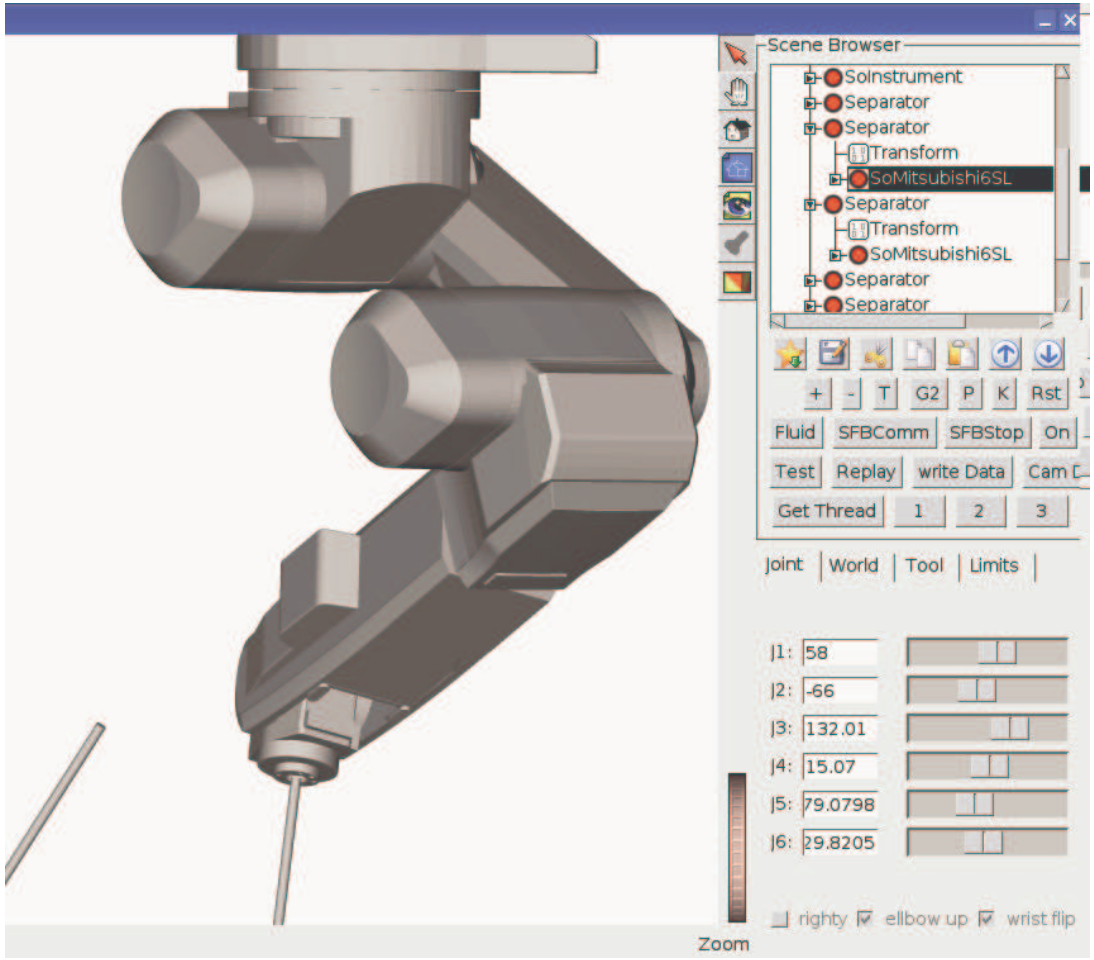
Scene Browser and Context Menus

- Szene wird durch eine Baumstruktur repräsentiert und in einem OpenGL-Fenster graphisch dargestellt
- Für jeden Knotentyp existiert ein Kontextmenü zur Einstellung der relevanten Parameter
- Änderungen in der Szene werden zuerst in der 3D Darstellung angezeigt und können danach in das reale System übernommen werden
- Mit dem „Scene Browser“ können Änderungen am Szenenbaum vorgenommen werden (einfache CAD-Funktionalität



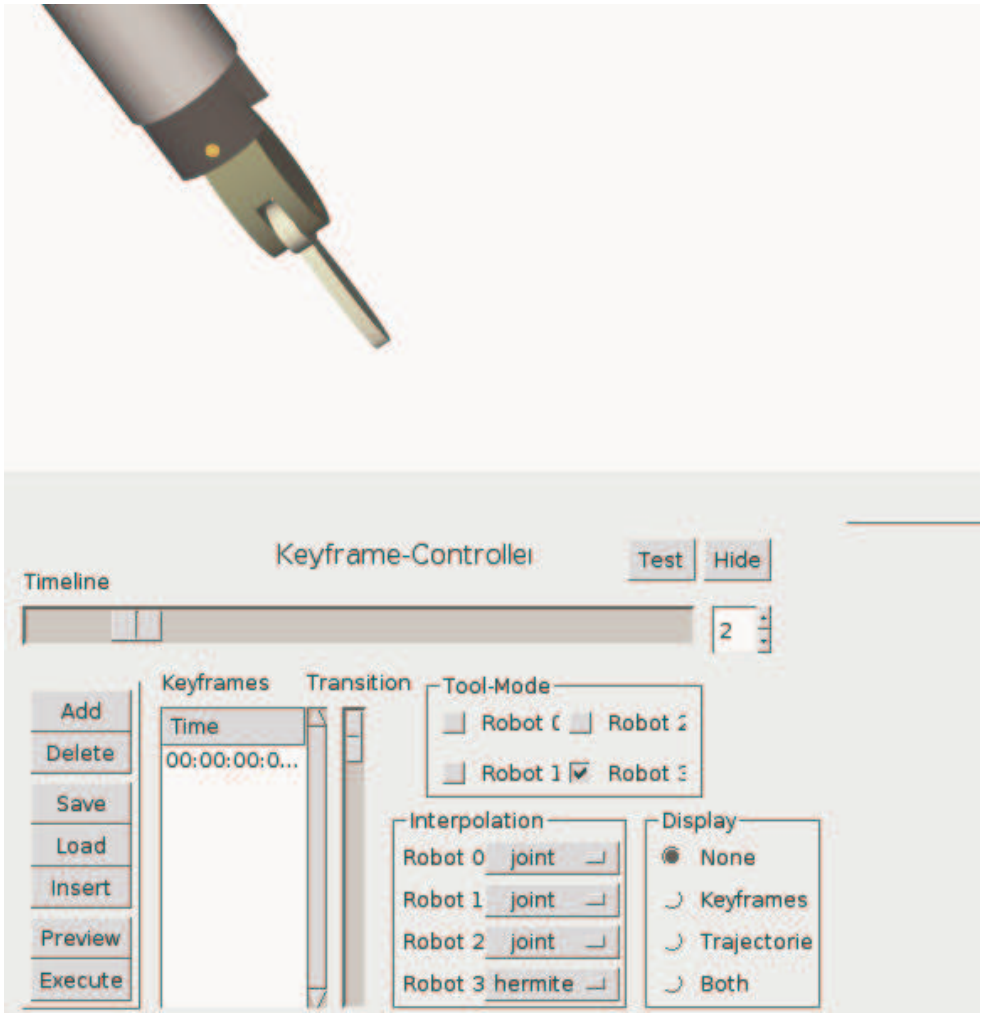
Scene Browser and Context Menus

- Szene wird durch eine Baumstruktur repräsentiert und in einem OpenGL-Fenster graphisch dargestellt
- Für jeden Knotentyp existiert ein Kontextmenü zur Einstellung der relevanten Parameter
- Änderungen in der Szene werden zuerst in der 3D Darstellung angezeigt und können danach in das reale System übernommen werden
- Mit dem „Scene Browser“ können Änderungen am Szenenbaum vorgenommen werden (einfache CAD-Funktionalität



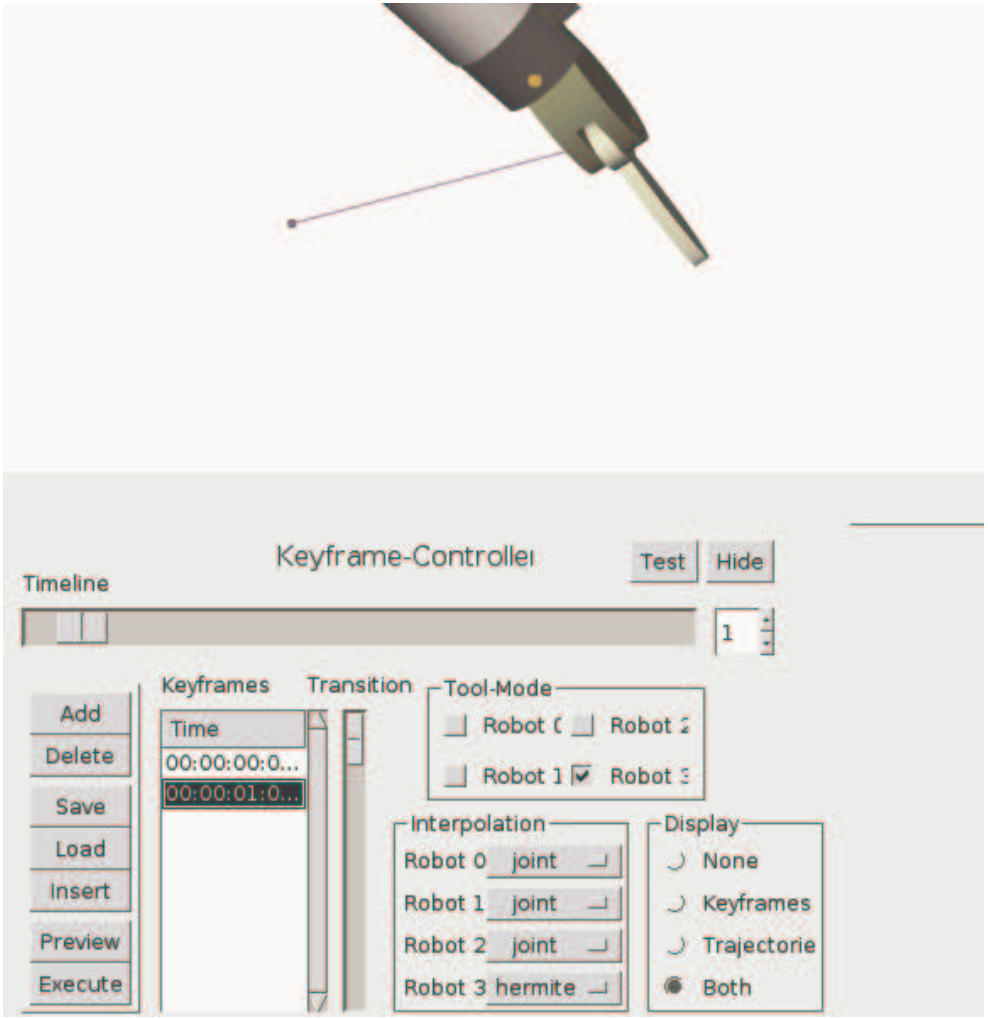
Animations-Interface

- Signifikante Positionen werden als “Key-Frames” abgespeichert (zusätzlich ein Zeitstempel für die spätere Synchronisation)
- Verschiedene Interpolations-möglichkeiten stehen zur Auswahl: Linear, Hermite, KB splines
- Die Parameter der “Key-Frames” können jederzeit angepasst werden, die Trajektorie wird entsprechend neu berechnet
- Die Trajektorie kann in der Simulationsumgebung vor ihrer Ausführung getestet werden



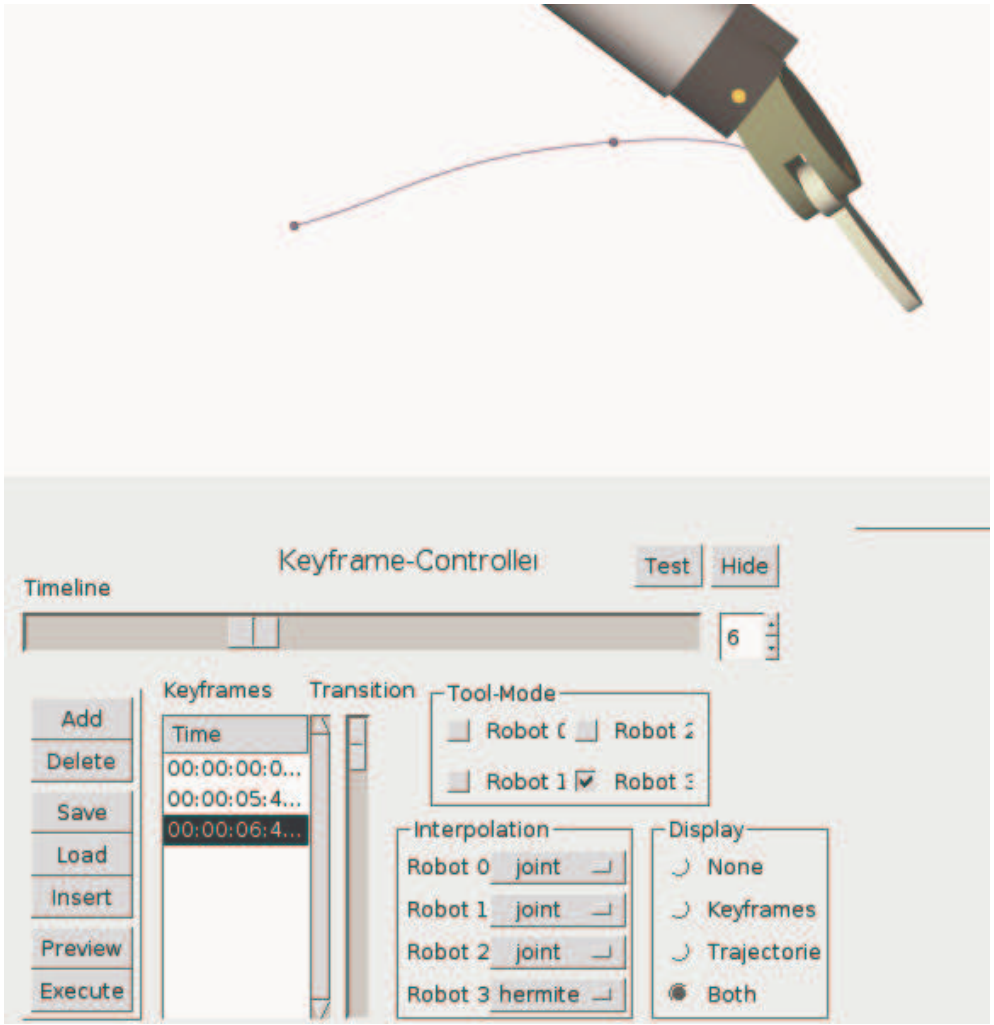
Animations-Interface

- Signifikante Positionen werden als “Key-Frames” abgespeichert (zusätzlich ein Zeitstempel für die spätere Synchronisation)
- Verschiedene Interpolations-möglichkeiten stehen zur Auswahl: Linear, Hermite, KB splines
- Die Parameter der “Key-Frames” können jederzeit angepasst werden, die Trajektorie wird entsprechend neu berechnet
- Die Trajektorie kann in der Simulationsumgebung vor ihrer Ausführung getestet werden



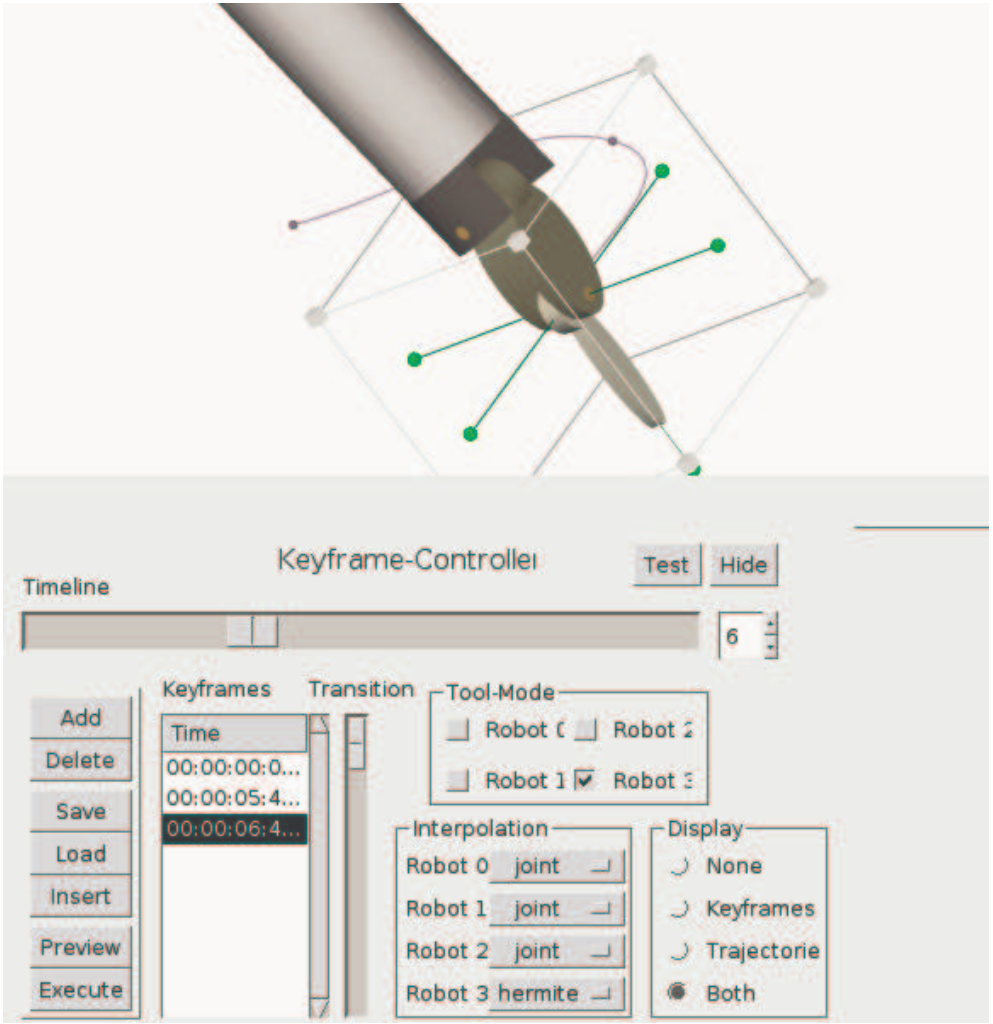
Animations-Interface

- Signifikante Positionen werden als “Key-Frames” abgespeichert (zusätzlich ein Zeitstempel für die spätere Synchronisation)
- Verschiedene Interpolations-möglichkeiten stehen zur Auswahl: Linear, Hermite, KB splines
- Die Parameter der “Key-Frames” können jederzeit angepasst werden, die Trajektorie wird entsprechend neu berechnet
- Die Trajektorie kann in der Simulationsumgebung vor ihrer Ausführung getestet werden

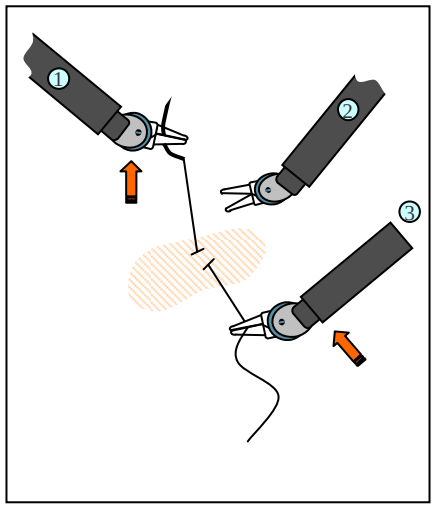


Animations-Interface

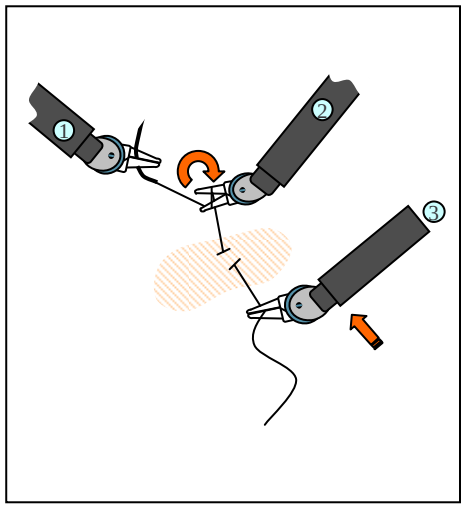
- Signifikante Positionen werden als “Key-Frames” abgespeichert (zusätzlich ein Zeitstempel für die spätere Synchronisation)
- Verschiedene Interpolations-möglichkeiten stehen zur Auswahl: Linear, Hermite, KB splines
- Die Parameter der “Key-Frames” können jederzeit angepasst werden, die Trajektorie wird entsprechend neu berechnet
- Die Trajektorie kann in der Simulationsumgebung vor ihrer Ausführung getestet werden



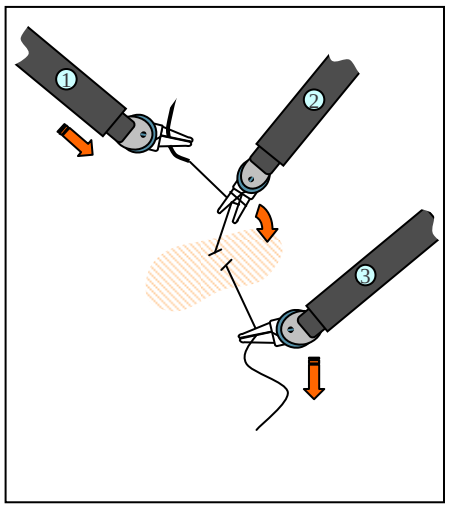
Automatisches Knoten (Programmiert)



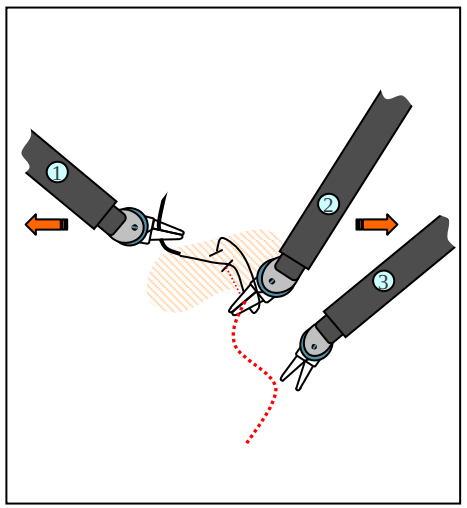
(1) herausziehen (3) spannen



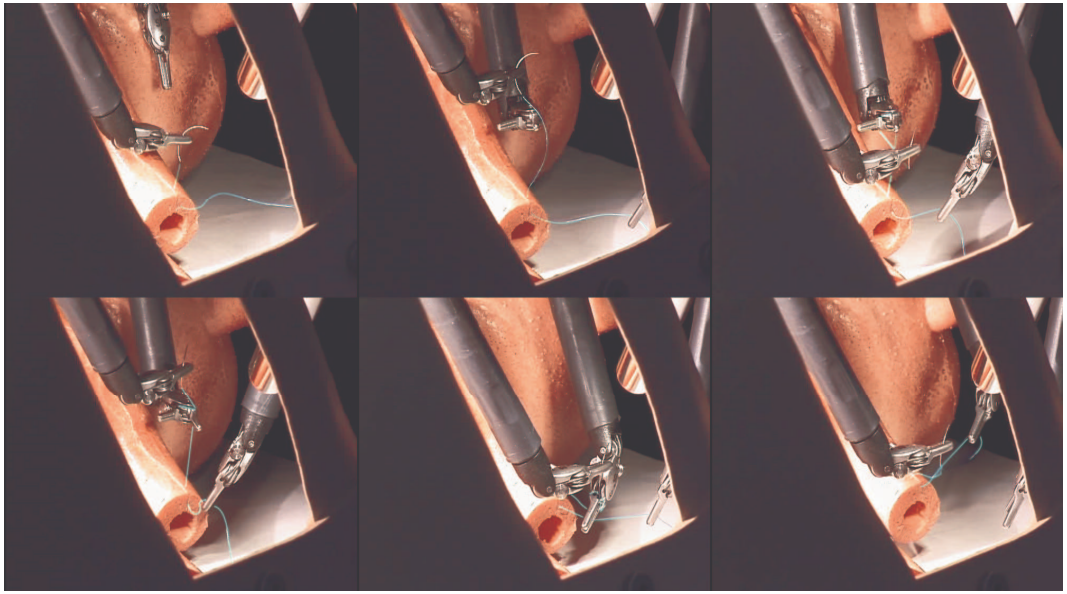
(1) wickeln (2) (3) spannen



(2) Ende greifen (1)+(3) spannen



Knoten fertig stellen



Automatisches Knoten (Programmiert)

Kamerasicht von außen

Sicht durch Endoskop

Skilltransfer: Arbeitsablauf

Dialog Mensch-Maschine am Beispiel des chirurgischen Knotens

„Ich zeige Dir wie man einen Knoten macht“

Neuer Skill: „Knoten“ anlegen

„Was ist ein Knoten?“

Faden herausziehen, rechten Greifer öffnen, um zweiten Greifer wickeln…

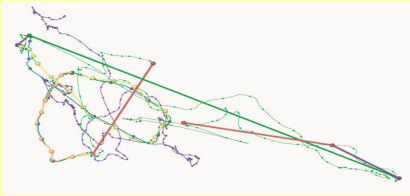
Abstraktes Muster für „Knoten“ anlegen:

Rechte Hand:

1. Linearbewegung bis Greifer geschlossen wird
2. 2D Primitiv bis Wendepunkt erreicht
3. Linearbewegung bis max. Kraft erreicht …

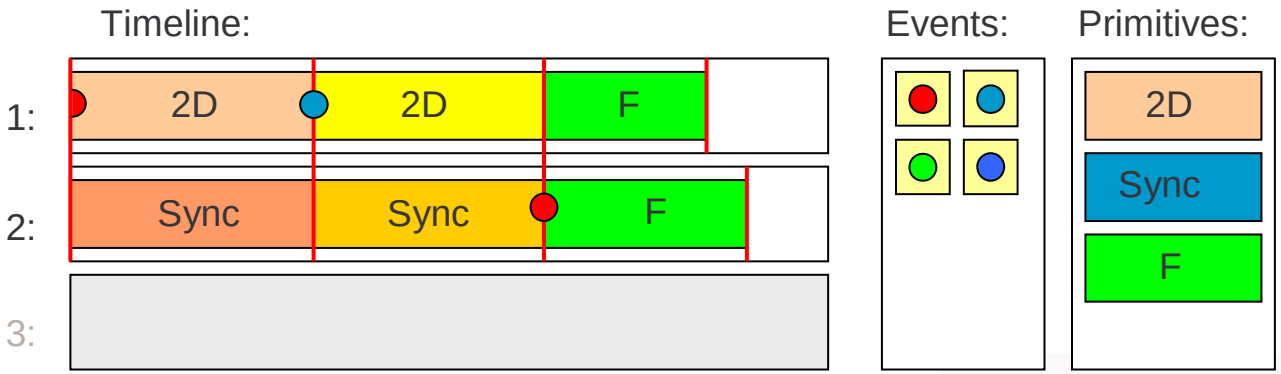
„Wie macht man einen Knoten?“

Vorführungen des Knotens



Extraktion konkreter Information (Geometrie, Zeit, Kraft …) aus 1 – n Demonstrationen

Skilltransfer (abstraktes Konzept + konkrete Vorführungen)

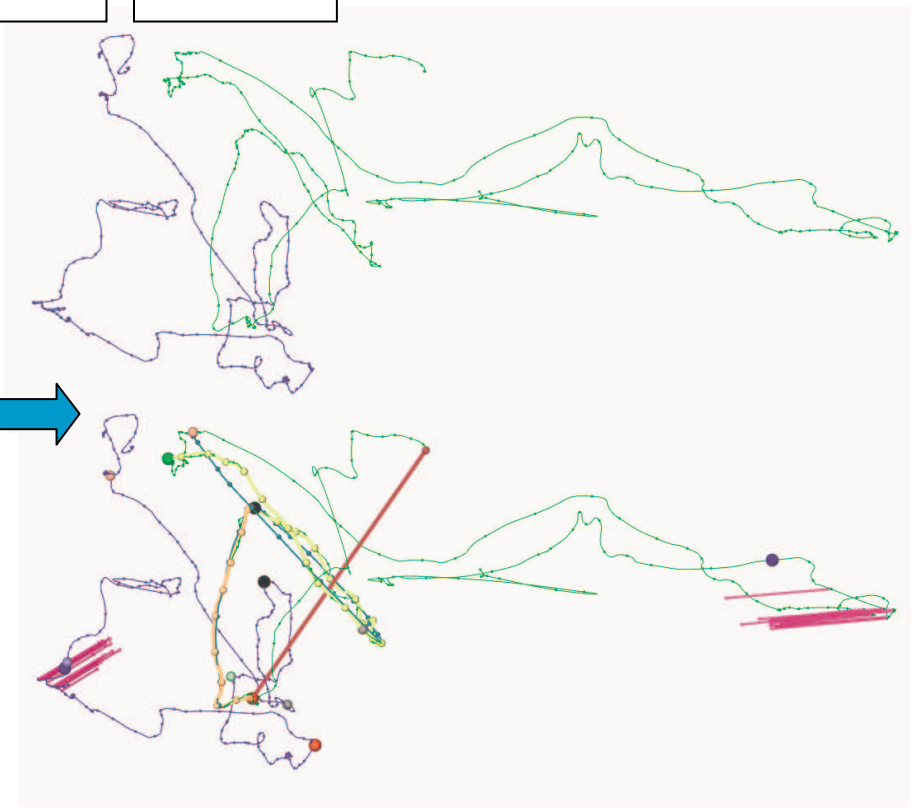


next = findChunk(0, EventType)

next = find2DMotion(next, EventType)

next = find2DMotion(next, EventType)

next = findForceControlledMotion(next, EventType)



Implementierung und Anwendung der gefundenen Primitiva

Anforderung: Bereits nach einer Vorführung muss eine robuste Ausführung der Skills gewährleistet sein (zumindest eine 1:1 Wiederholung).

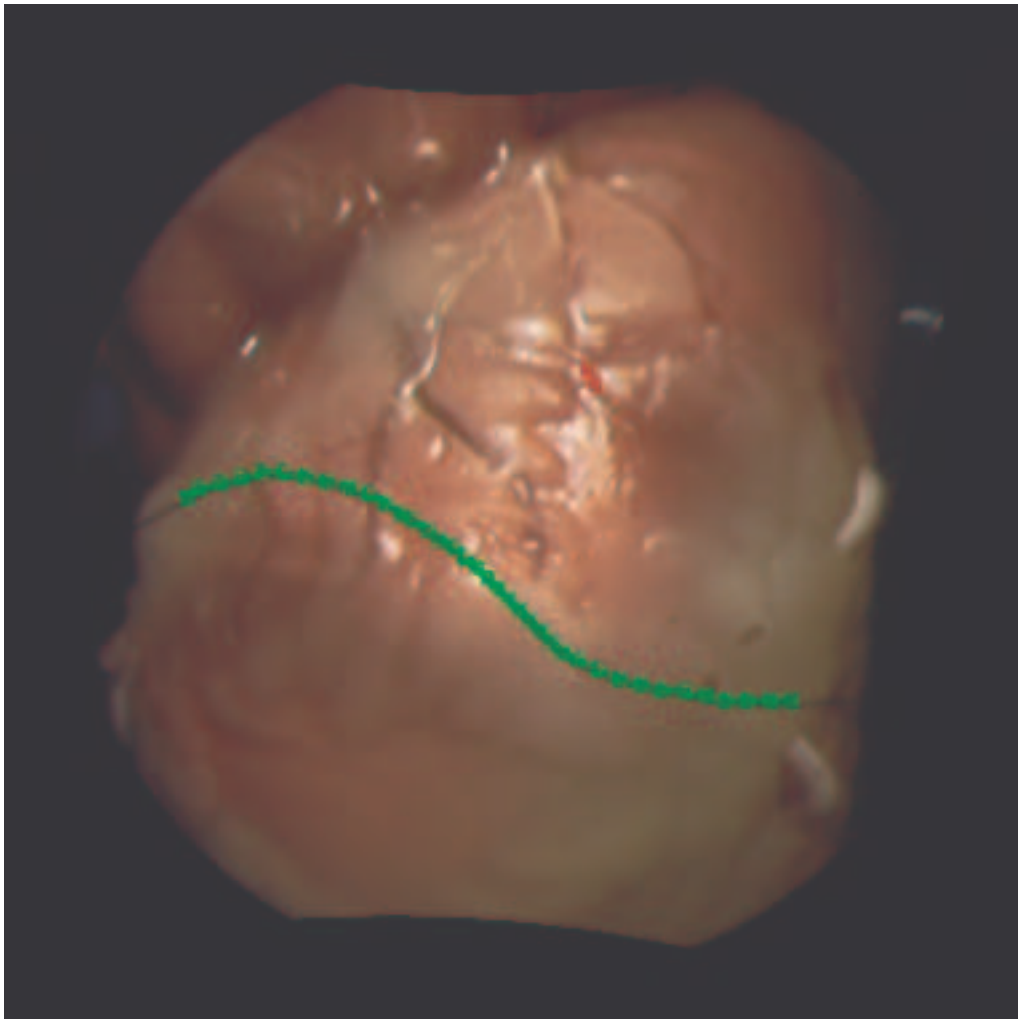
Verbesserung: Durch Integration von Zusatzwissen kann eine adaptive Ausführung in unbekannter Umgebung bereits nach wenigen Vorführungen (im Extremfall einer) erfolgen.

z.B.: Festlegen des Zielpunkts für Linearbewegung mit **Fadenerkennung**

z.B.: Adaptive 2D-Bewegung mit Hilfe von **Fluid-Dynamik**

z.B.: Synchronisierte 2D-Bewegung mit Hilfe von **neuronalen Netzen**

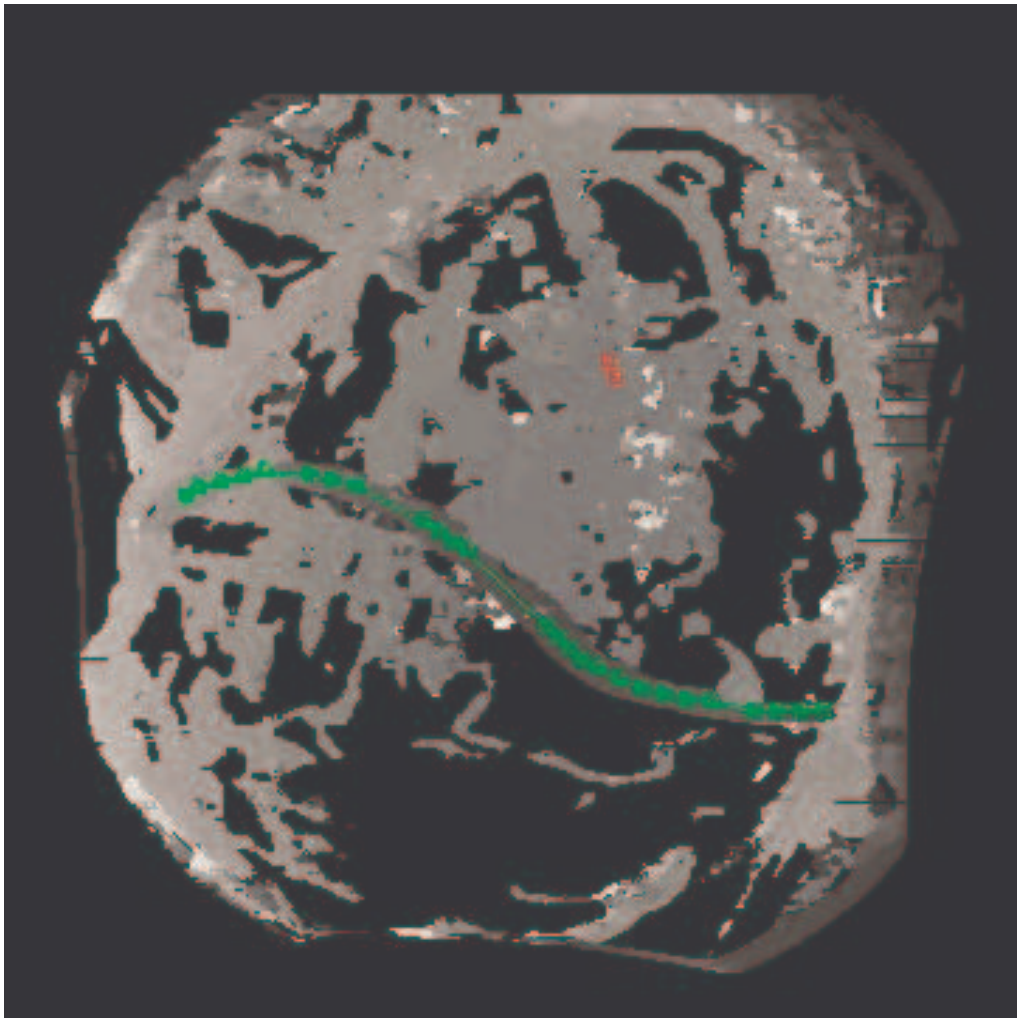
Rekonstruktion des Operationssitus



Fadenerkennung

- Kantenfilter unter Berücksichtigung der Strukturbreite und des Kontrastes
- Farbfilter um falsche Zuordnungen zu eliminieren

Rekonstruktion des Operationssitus



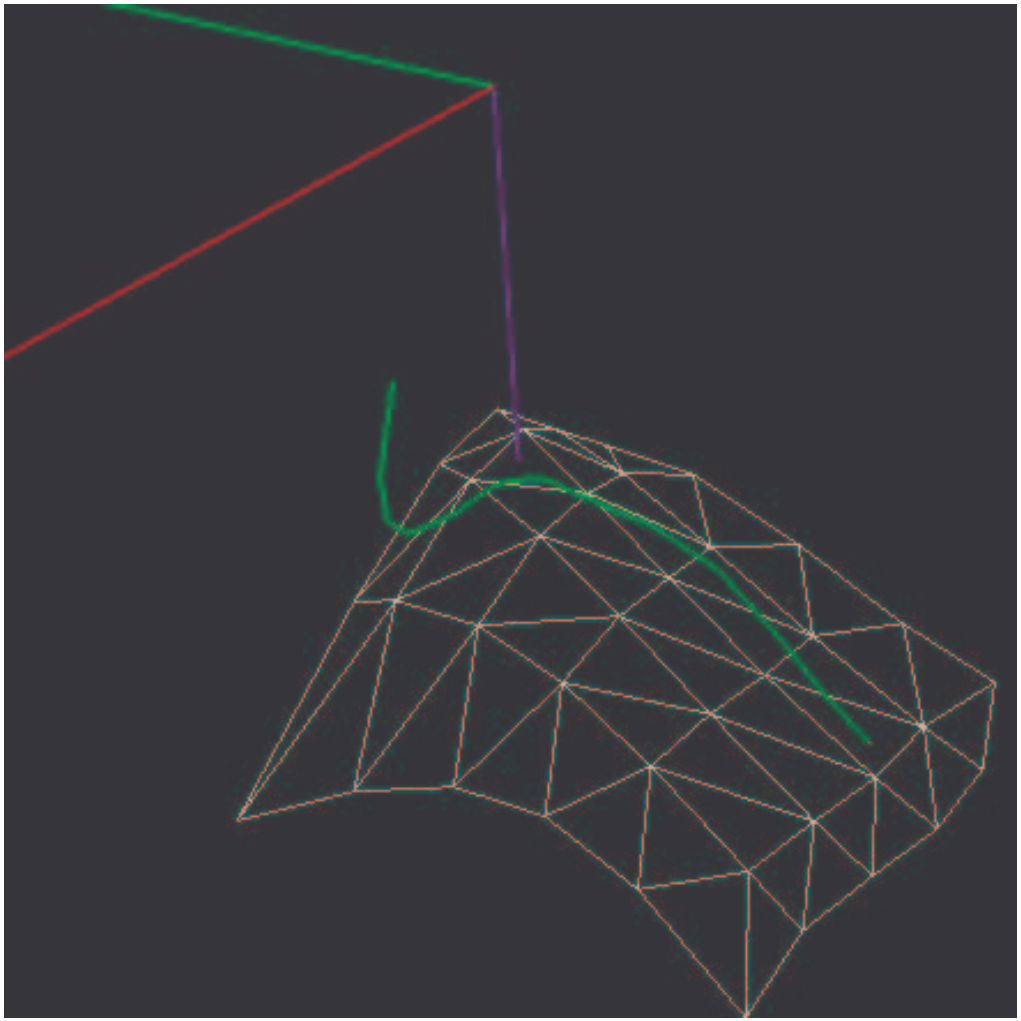
Fadenerkennung

- Kantenfilter unter Berücksichtigung der Strukturbreite und des Kontrastes
- Farbfilter um falsche Zuordnungen zu eliminieren

Disparitätskarte

- Punkte entlang des Fadenverlaufs mit guten Korrespondenzwerten auswählen
- Oft lückenhaft

Rekonstruktion des Operationssitus



Fadenerkennung

- Kantenfilter unter Berücksichtigung der Strukturbreite und des Kontrastes
- Farbfilter um falsche Zuordnungen zu eliminieren

Disparitätskarte

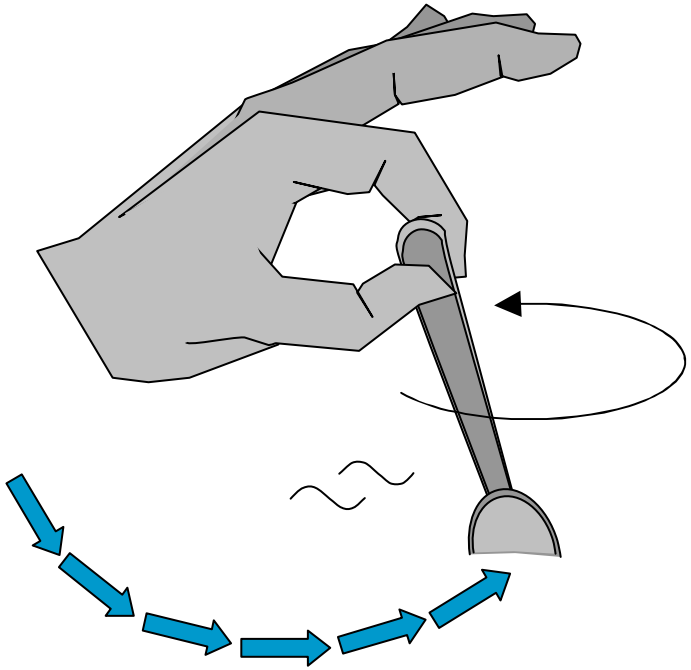
- Punkte entlang des Fadenverlaufs mit guten Korrespondenzwerten auswählen
- Oft lückenhaft

3D-Rekonstruktion

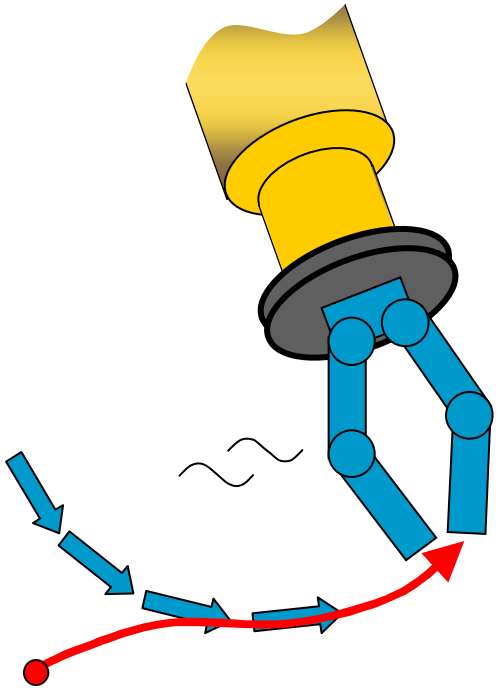
- Kubische Spline für den Faden
- Interpolierender NURBS-Patch für den Hintergrund

Anwendung von 2D Primitiven

Grundlegende Idee:

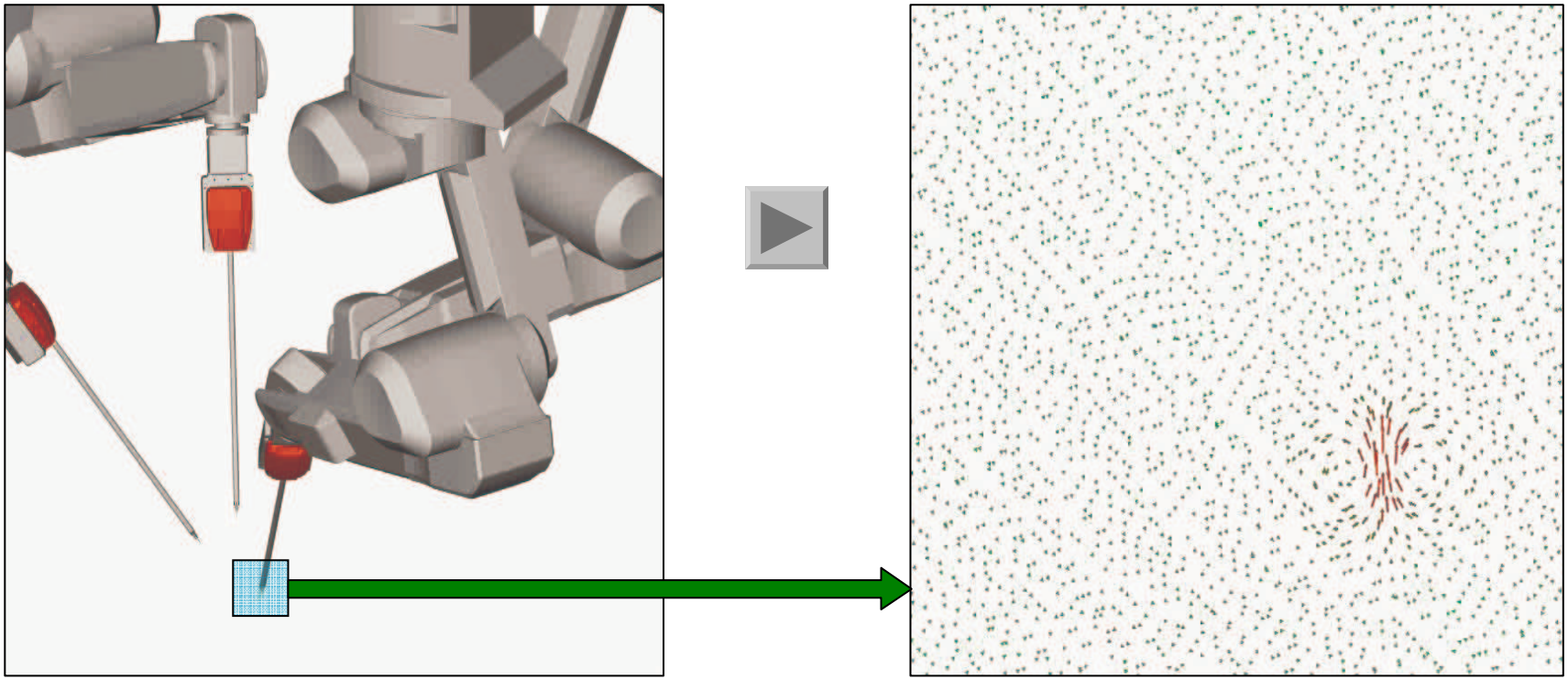


Vorführung durch Benutzer:
Umrühren einer Flüssigkeit



Nachahmung durch Roboter:
Partikel in Flüssigkeit werfen

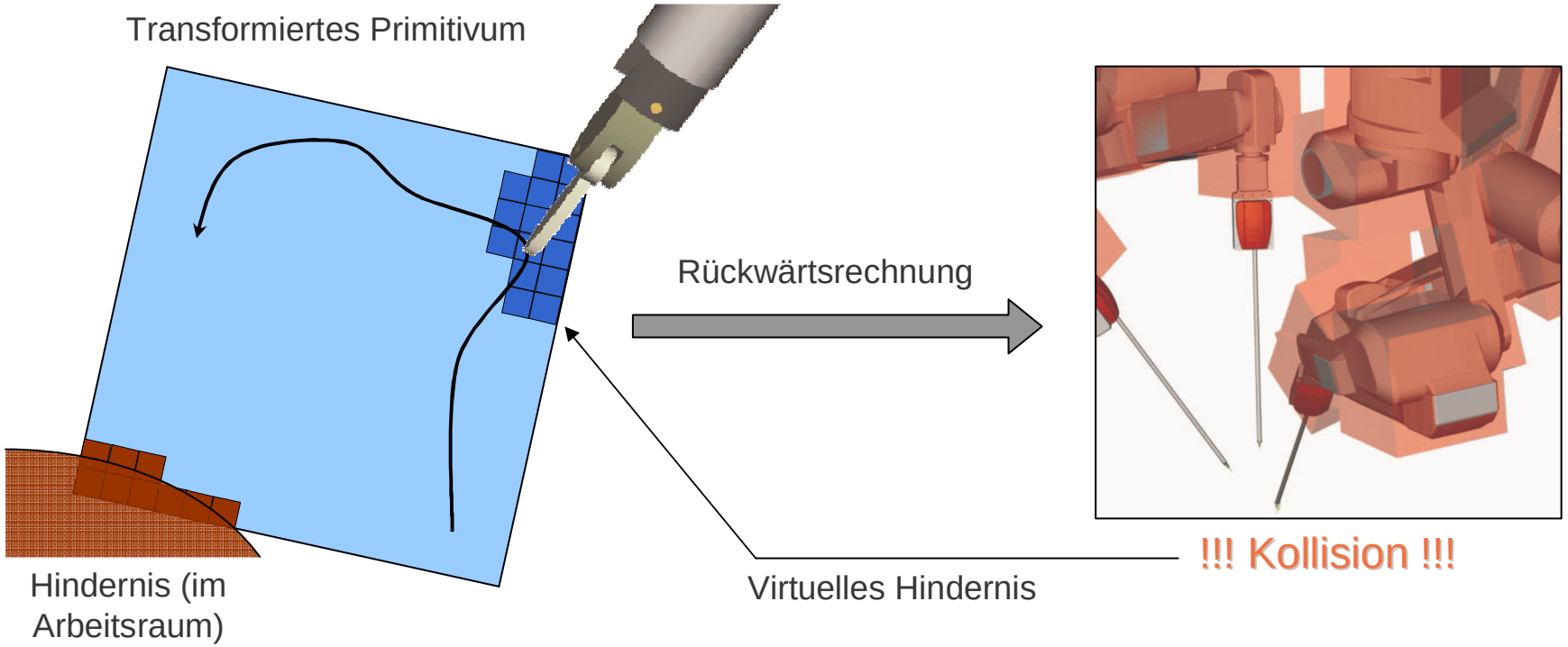
Anpassung eines Primitivums an neue Umgebung



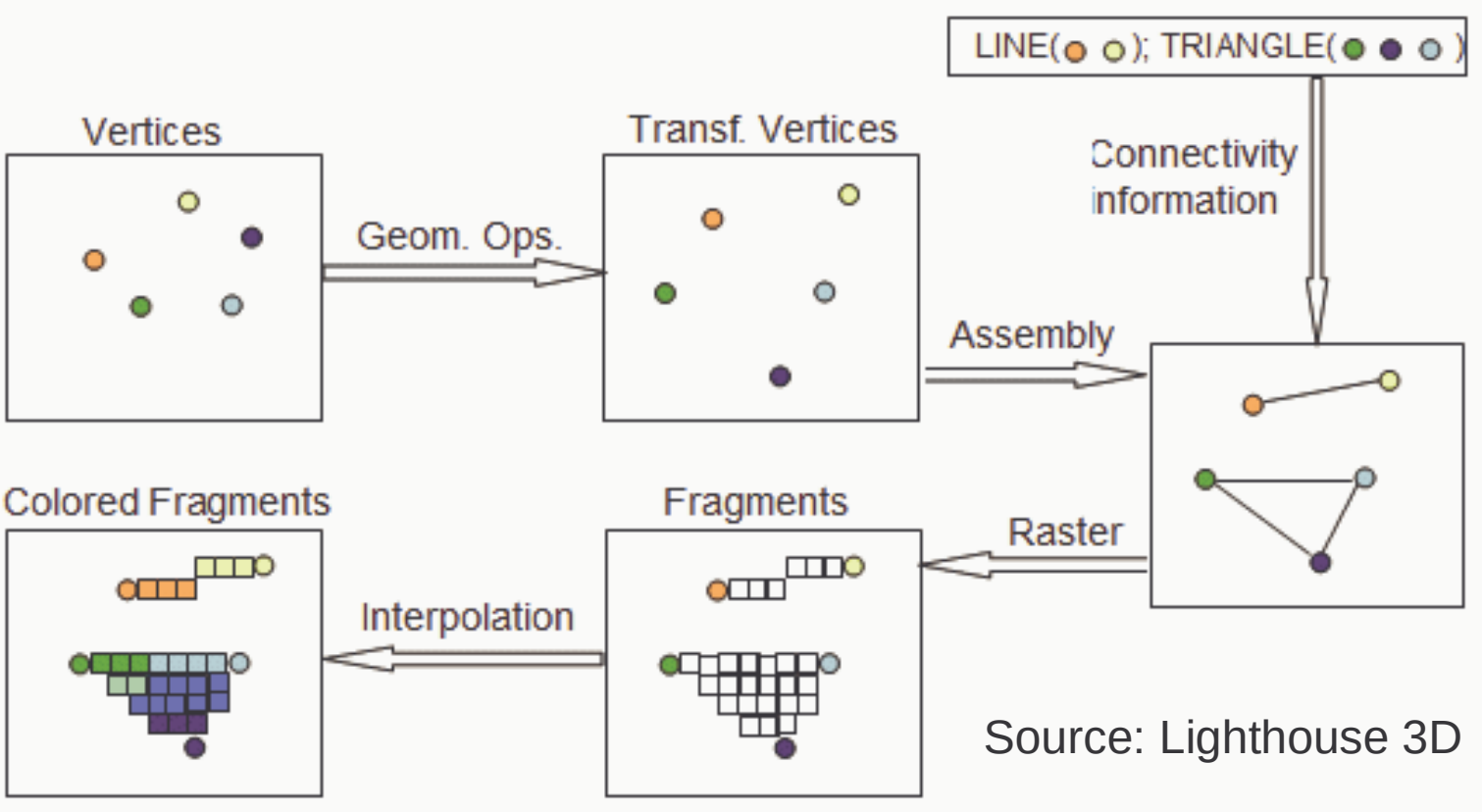
- Primitiv (d.h. die erzeugte Strömung) wird in die neue Umgebung transformiert
- Problem: Es können Kollisionen der Endeffektoren bzw. Roboter auftreten

Kollisionsvermeidung

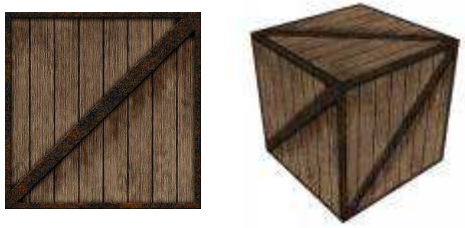
Lösung: Kollisionen der Endeffektoren werden direkt als Hinderniszellen implementiert.
Kollisionen der Roboter werden in virtuelle Hindernisse umgesetzt.



Für jede Zelle der Fluidsimulation: Roboter auf Kollisionen prüfen
Kollisionserkennung auf **Subzell-Level** (Faktor 5) zur Weiterverarbeitung
z.B. für obige Anwendung: prüfe 250 x 250 Positionen: **hoher Rechenaufwand**

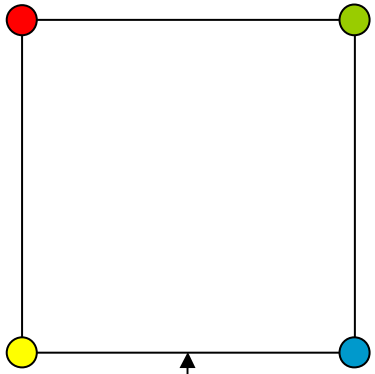


Textur mit Fragment-Shader applizieren

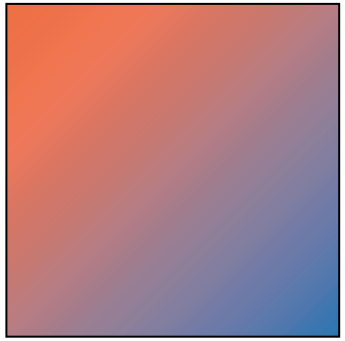


GPU Rückwärtsrechnung und Kollisionserkennung

Rechteck mit OpenGL zeichen



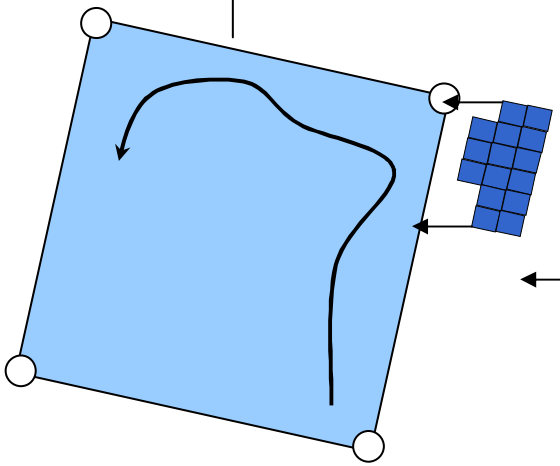
Vertex Shader
+ Rasterization
+ Interpolation



Fragmet Shader 1:
Rückwärtsrechnung
Input: Interpolierte Farben als Koordinaten
Output: Roboterwinkel in Textur schreiben

Fragmet Shader 2:
OBB Kollisionserkennung
Input: Roboterwinkel von Textur lesen
Output: Kollisionen im Framebuffer speichern

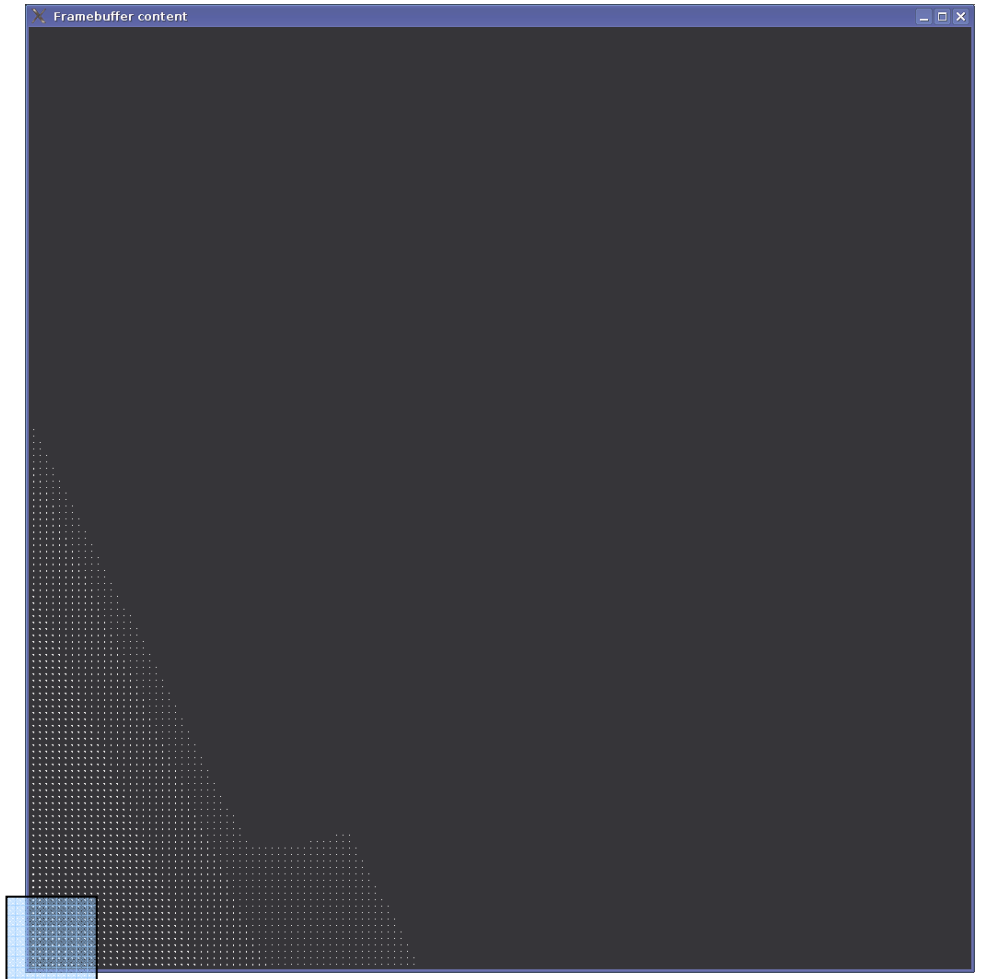
3D Koordinaten als Farb-
information interpretieren



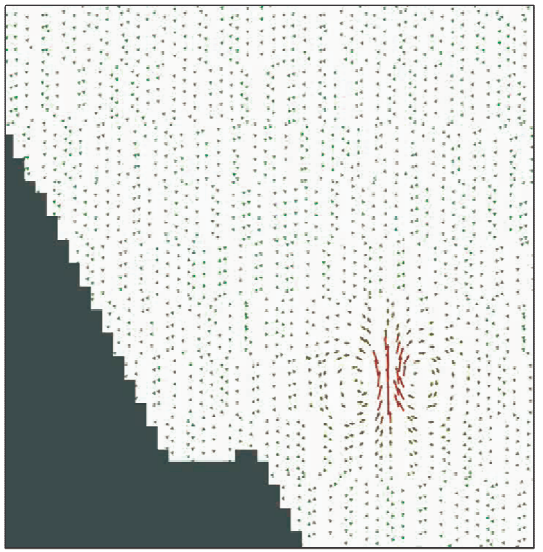
Hinderniszellen vom
Framebuffer auslesen

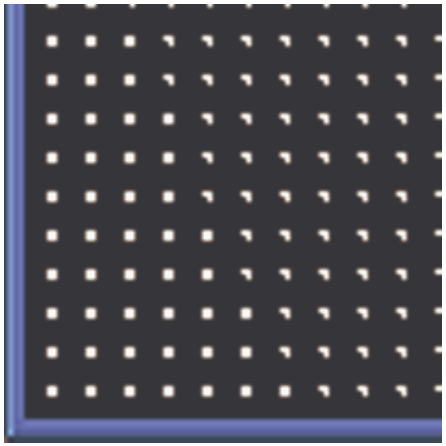
Ergebnis

Inhalt des Framebuffers

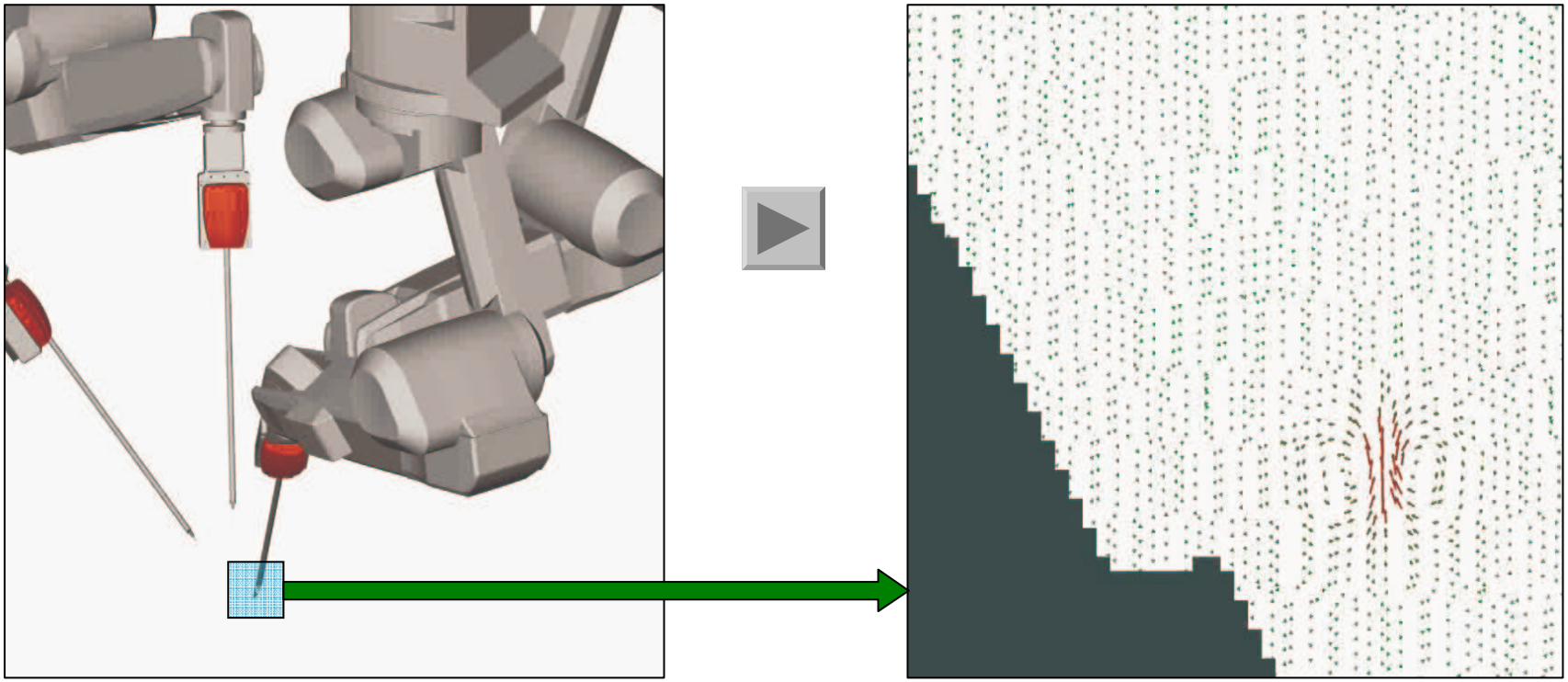


Ausgelesene Hindernisse





Anpassung des Primitivs mit Kollisionserkennung



- Simulationumgebung wird durch Hinderniszellen begrenzt mit entsprechend gesetzten Randbedingungen
- Stromlinien können sich nie schneiden oder kollidieren

Zusammenfassung und Diskussion

- Generalisierung von Bewegungen mit Hilfe von Fluid-Dynamik
  nach nur einer Vorführung

- Deutliche Beschleunigung der Kollisionserkennung mit GPU computing
  14.1 sek. auf Athlon64 1600 MHz vs. 1.36 sek. auf NVIDIA 6600 GPU

- Probleme: Fluidsimulation muss rechtzeitig gestoppt werden
  (kann mit Hilfe von Ausströmbedingungen erreicht werden)
  Hoher Rechenaufwand für Fluid-Dynamik: bis zu 30 sek. für
  die Anwendung eines Primitivums: aber gut parallelisierbar
  Uneinheitliche Grafikhardware: Unterschiedliches Verhalten von Shader
  Programmen: z.B. unzureichende Präzision von atan2
  Einschränkungen der Grafikhardware: der Speicher der momentan
  eingebauten Karte ist auf 4500 Assemblerbefehle beschränkt: →
  separate Shader für Rückwärtsrechnug und Kollisionserkennung
  Texturgröße muss eine Zweierpotenz sein