6th European PhD School in Robotic Systems
—
Kowledge-driven task specification and execution

Herman Bruyninckx
University of Leuven
Eindhoven University of Technology

Leuven, August 4–8, 2014

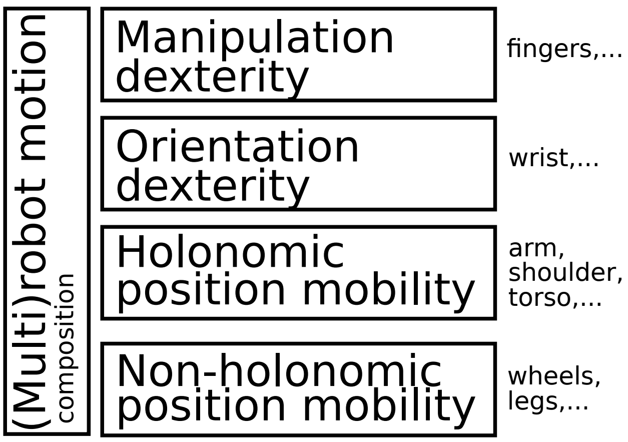
Overview

- mainstream approach "Sense-Plan-Act"
- advocated approach "Task-Skill-Motion"

⇒ motivations: **composability** of constraint-based task specification, and **composability** of "model-to-model transformations" from specification to executable software.

- simplest case: motion specification and execution for composite robot platform: the "5F" **physics** provides the constraints.
- generic case: task specification and execution for the composition of robot platform motion capabilities with:
  - "5A": levels of abstraction
  - "5T": task levels
  - "5G": geometry for world model
  - "5K": knowledge (task, world, affordance, common knowledge)
  - "5COP": constraint-optimization problems

5F — Robot motion functionality



5A — Levels of abstraction

Abstraction levels
composition

Mereology	*symbolic* "contains" relationships ("parts'"-"whole")
Topology	*symbolic* "connection" relationships
Discrete	*discrete* "event-centric" selection of activity relationships
Space-time	*continuous* time and space relationships

5T — Task levels

Mission
Series of Tasks to achieve a purposeful end-result
(*symbolic* composition)

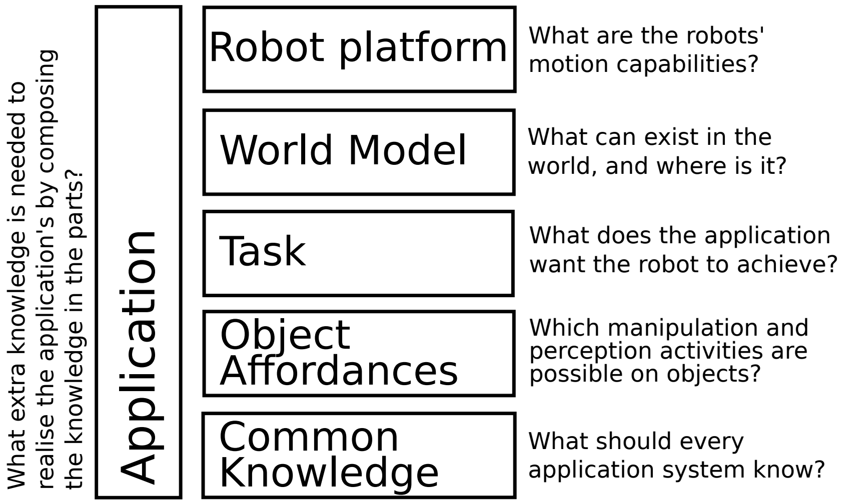
Task	*series* of *activities* performed on *objects* (*discrete* world)
Object	transformation of sensor/actuator features into *object features* (object-level *continuous* world)
Feature/setpoint	transformations between real world and software representation (*virtual continuous* world)
Sensor/Actuator	direct interactions with the real world (*real continuous* world)

5G — Geometry for World model

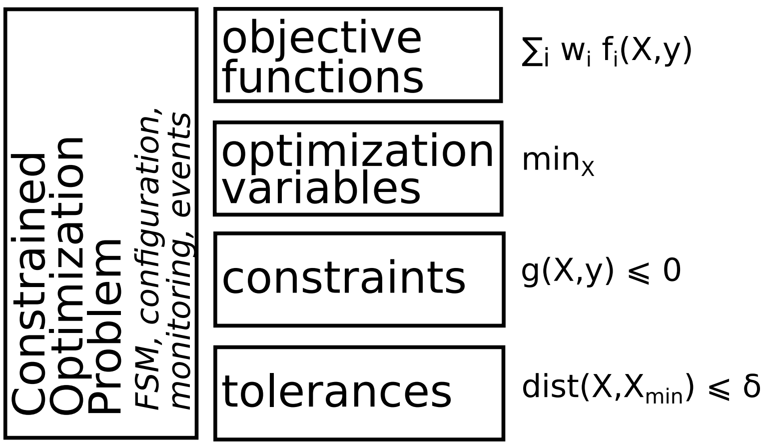
Mesh/Grid
composition
E.g.: OGC NetCDF, XDMF

Cell/Volume	3D; octree, frame
(Sur)Face	2D; half plane, polygon
Line/Edge	1D; line, interval
Point/Vertex	0D; point, point cloud

5K — Knowledge relationships



5COP — Constraint-optimization problem



5COP — Constraint-optimization problem (2)

- figure on previous slide:
  - shows only the continuous version.
  - there are *dozens* of different "continuous versions".
  - requiring dozens of different "solvers"!
- at discrete level:
  - *discrete/hybrid event systems* optimization, *timed automata*,...
  - each "state" solves a continuous-domain COP.
- at symbolic level:
  - *constraint satisfaction*, *theorem provers*,...
  - = find "primitives" that "solve"/"satisfy" given constraint

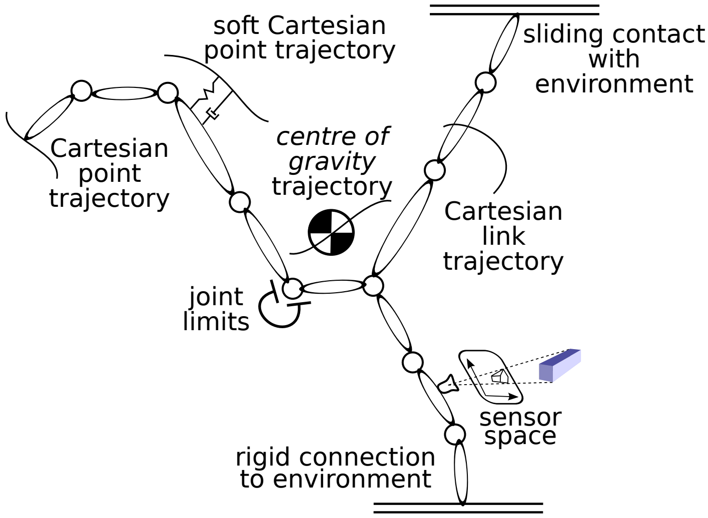
Challenges:

- formulation, reposity, query, bookkeeping,... of all such constraints!
- solvers, solvers, solvers!
- predictabilty of composition of constraints and solvers!

From "Sense-Plan-Act" to "Task-Skill-Motion"

- **Sense-Plan-Act** pipelines:
  - no "Quality of Service" interactions between Sense, Plan and Act sub-systems → no runtime adaptation.
  - deployed in heavy-weight "nodes"/"components" → context switch overhead, data distribution overhead
  - no constraints on Sense & Plan by Act
    no constraints from object affordances on Sense, Plan or Act
- **Task-Skill-Motion** networks:
  - "Every application is a constraint optimization problem!"
  - software architecture based on the Composition Pattern allows to realise:
    - constraints on Sense & Plan by knowledge about Act
    - constraints from world model and object affordances on Sense, Plan and Act

Start with "robot platform": where is the constraint optimization?



Intermezzo: MDE for constraint optimization

M3:

task state & domain	$X \in \mathcal{D}$
desired state	X_d
robot state & domain	$q \in \mathcal{Q}$
objective function	$\min_q f(X)$
equality constraints	$g(X) = 0$
inequality constraints	$h(X) \leq 0$
tolerances	$d(X, X_d) \leq A$
solver	**algorithm** computes q

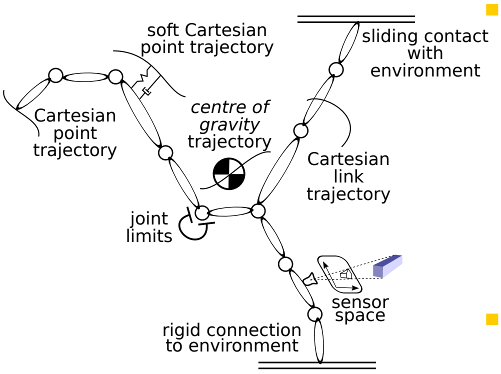
M2:

- fill in types for f, X, q for a particular domain, a particular "robot", and a particular type of solver
- applies to symbolic, discrete and continuous domains!
- applies to 5R, and "sensor space"!

M1:

- fill in parameter values for f, X, ...
- fill in solver implementation.

Back to the "robot platform"…



- **Objective functions**:
  - energy: mechanical, electrical, process,…
  - posture: comfortable behaviour near limits, singularities,…
  - time
  - "trajectory"-specific
  - weighting or priorities
- **Constraints**:
  - contact force interval
  - speed interval
  - impedance interval
  - "trajectory"-specific
  - …
- **Tolerances**:
  - on constraints and objective functions
  - when is "optimal" good enough?

Solved!!

"The" solution is 40 years old…

Е. П. ПОПОВ, А. Ф. ВЕРЕЩАГИН,
С. Л. ЗЕНКЕВИЧ

МАНИПУЛЯЦИОННЫЕ
РОБОТЫ

ДИНАМИКА И АЛГОРИТМЫ

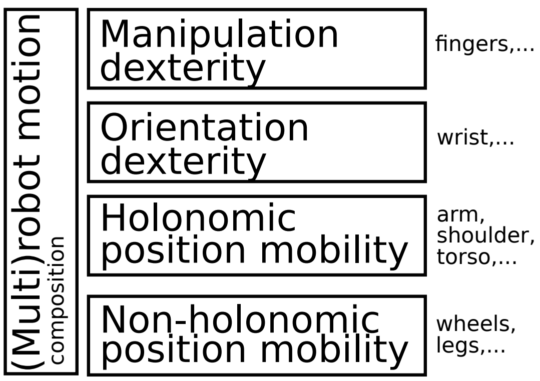
МОСКВА «НАУКА»
ГЛАВНАЯ РЕДАКЦИЯ
ФИЗИКО-МАТЕМАТИЧЕСКОЙ ЛИТЕРАТУРЫ
1978

Russian textbook from 1978, including Algol code:

- Popov, Evgenii Pavlovich; Vereshchagin, Anatolii Fedorovich; and Zenkevich, Stanislav Leonidovich, *Manipulyatsionnye roboty: dinamika i algoritmy*, Nauka, Moscow, 1978.

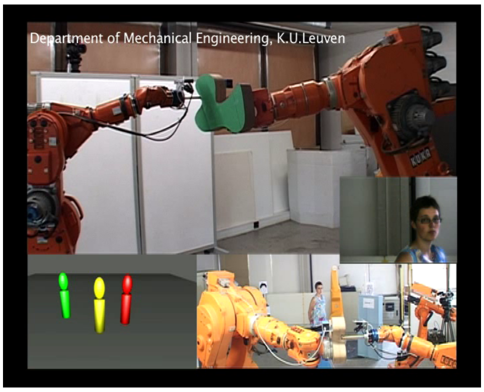
Approach: acceleration-constrained mechanical motion, via Gauss Principle of Least Constraint

"The" solution is 40 years old… (2)



- any tree structure in kinematic chain allowed!
- constraints on joints **and** links allowed!
- all constraints (holonomic, non-holonomic; priorities, weighting) are composable at acceleration level!

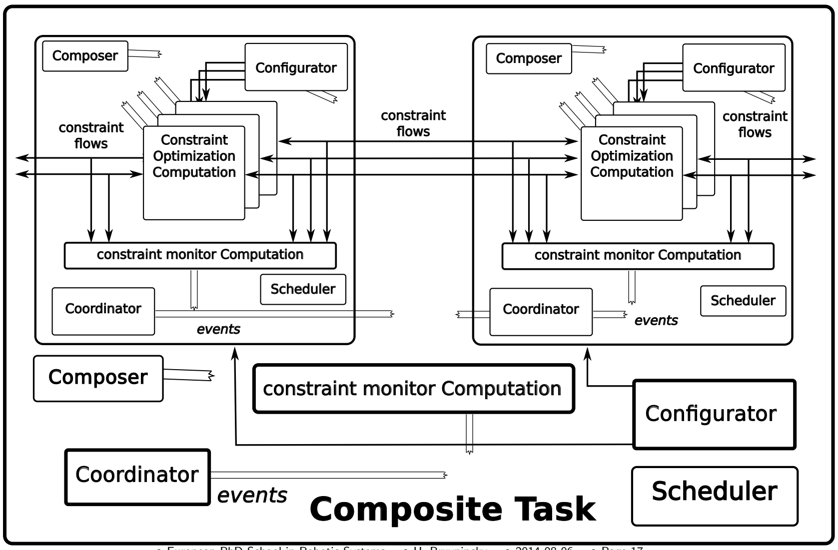
What does "Solved!!" mean. . . ?



Observations:

- these are "just" instantaneous optimization solutions!
- but they already include **sensori**-motor control:
  "Bayesian networks" are constraint optimization solvers!

What does "Solved!!" mean. . . ? —Task Composition Pattern—



What does "Solved!!" mean. . . ? —Task Composition Pattern (2)—

Pattern is a knowledge/information **model**, and not necessarily a software **architecture**:

- deployment on SW architecture can take other software **resource constraints** into account: shared memory, scheduling of processes, middleware constraints,. . .
- subsequent deployment on HW architecture can take other hardware **resource constraints** into account: communication bandwidth and topology, access to servers "in the cloud", available RAM and CPU, energy,. . .

But since it is driven by "constraint optimization", the pattern applies equally well to these deployments.
(See *Software* lecture tomorrow.)

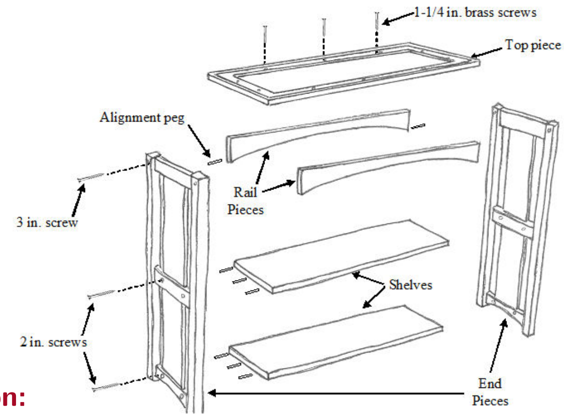
What does "Solved!!" mean. . . ?
—Task Composition Pattern (3)—

Pattern conforms to "5A" (composition of various levels of abstraction):

- each has different formalization and solvers;
- the composition takes place via:
  - constraint "flows" between "solvers".
  - Coordination & Configuration of "lower" levels' COPs by "higher" levels' knowledge.
  - events pierce "information hiding" boundaries to "lower" and "higher" levels
  - "names" of events and states at discrete level ground semantic meaning between continuous and symbolic levels.

⇒ the Pattern provides a (the. . . ?) MDE solution for "knowledge-driven, hierarchical control"

Now consider the full "5K". . .
Example: human-centric task specification



Discussion:

- what is the required *implicit knowledge*?
  (object affordances? platform capabilities?. . .)
- how to make it *explicit*?

Task-Skill-Motion: meta meta model

- **Task**:
  - constraints specification with **only task-dependent objects**
  - e.g., furniture assembly; filling bottle with water;. . .
- **Skill**: adds knowledge and solvers to cover
  - "affordance" execution capabilities of **platform type**.
  - **sensori**-motor integration → extended constraint optimization.
  - e.g., dual arm 7-DOF robots, contact detection, visual servoing,. . .
- **Motion**: adds
  - Quality of Service constraints of concrete platform **instances**: ABB Frida, 2×KUKA LWR, Qinetic gripper, force sensor,. . .
  - extra (partial) order on actions in task specification.
- **Software architecture**: deploys specification in **Activities**
  - constraint optimization **solver(s)** → "control" setpoints
  - perception, planning, learning,. . . **algorithms**.

⇒ **"Application"** composes all four together in deployable packages

Task-Skill-Motion: meta modelling

Our suggestions:

- fill in the class hierarchy knowledge of:
  - 5T ("Task specification"),
  - 5F ("Robot Platform"),
  - 5G ("World model"),

  with all 5A levels (symbolic, discrete, continuous) and "Common Knowledge" present in each class.
- create the "Object affordances" as sets of cross-links ("factual knowledge") between these class hierarchies.

($\Rightarrow$ explicit hiearchy in the 5K composition!)

- create an "Application" as set of cross-links ("factual knowledge") between, and Configurations of, all of the above, encoded as a "Task-Skill-Motion" class hierarchy.

Factual knowledge is where the money is!...

Discussion: feasibility? (dis)advantages? alternatives?...

Conclusions

- constraints are composable
- composition is not always predictable
- knowledge = constraints
- skill = factual knowledge about which constraints to apply for which task objective functions
- constraints apply across all "composition boundaries" of levels of abstraction, all sensori-motor control, all hybrid-event compositions,...

$\Rightarrow$ constraint-based task specification & execution are here to stay!

$\Rightarrow$ don't waste any more public money (or your time...) on the old-school Plan-Sense-Act :-)

Homework

- find task specifications for the furniture assembly mission