On the Design of Symbolic-Geometric Online Planning Systems

Lavindra de Silva - University of Nottingham
Felipe Meneguzzi - PUCRS

Motivation

- Programming autonomous robots is hard
  - Wide variety of algorithms
  - Varying granularities for data and decision-making
  - Implementations often combine **symbolic** (high-level) and **geometric** (low-level) reasoning
- Recent work on integrating symbolic and geometric planning

Background

"Classical" Planning

- Classical Planning: Discrete states (logic formulas) + atomic actions
- Problems are defined in terms of a domain, an initial state and
  - STRIPS/PDDL — declarative goal state
  - HTN — procedural desired task
  - HGN — hybrid between STRIPS/HTN
- Solution is a sequence of discrete **actions**

Geometric Planning

- At the lowest level, involves motion planning
  - 3D perception, search in continuous high-dimensional space
  - May include preferences and other high-level reasoning
- Environment comprises a 3D world with polygonal obstacles
- Solution is a collision-free sequence of **poses**

BDI Logic

- Originally proposed by Rao and Georgeff, later formalised in the AgentSpeak(L) language, assumes an agent

$$Ag = \langle Ev, Bel, PLib, Int \rangle$$

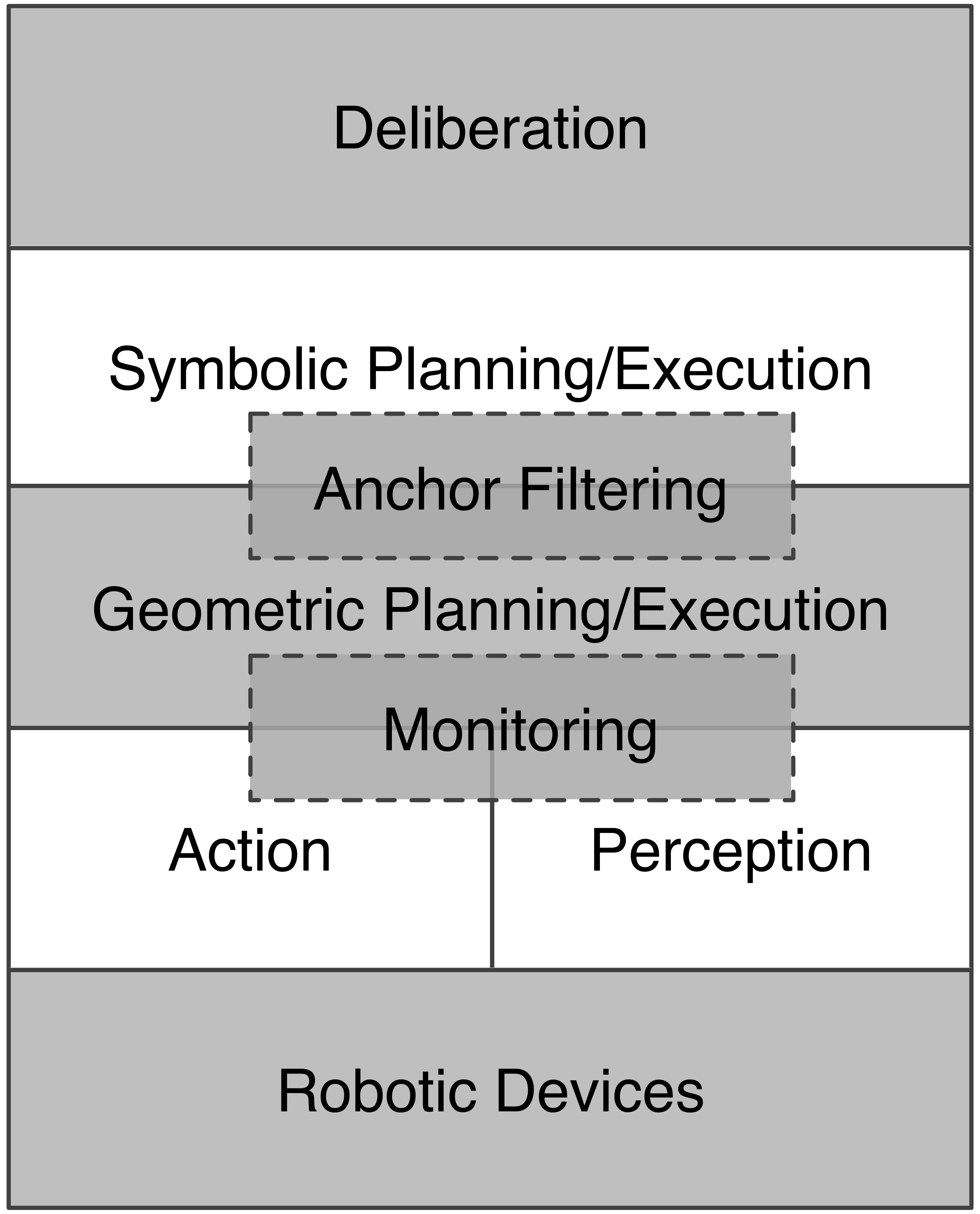
- Behaviour defined in terms of plan rules
  triggering_event : context <- body.

- where:
  - the triggering event denotes the events that the plan is meant to handle;
  - the context represent the circumstances in which the plan can be used;
  - the body is the course of action to be used to handle the event if the context is believed true at the time a plan is being chosen to handle the event.

Desiderata

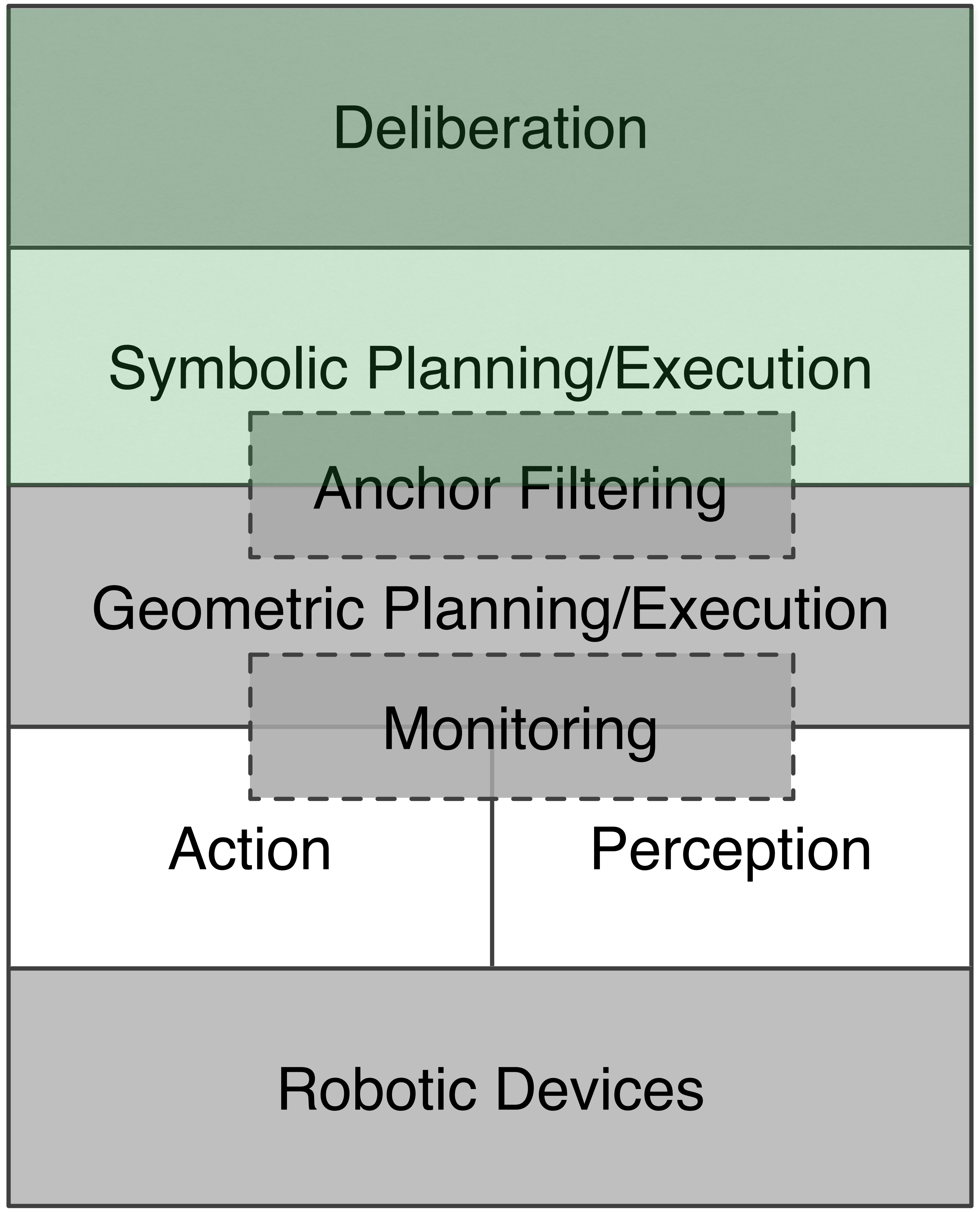
Abstract Architecture

- Robot behaviour implementation is often decomposed at various levels of abstraction
- We envision a tiered architecture incorporating advances



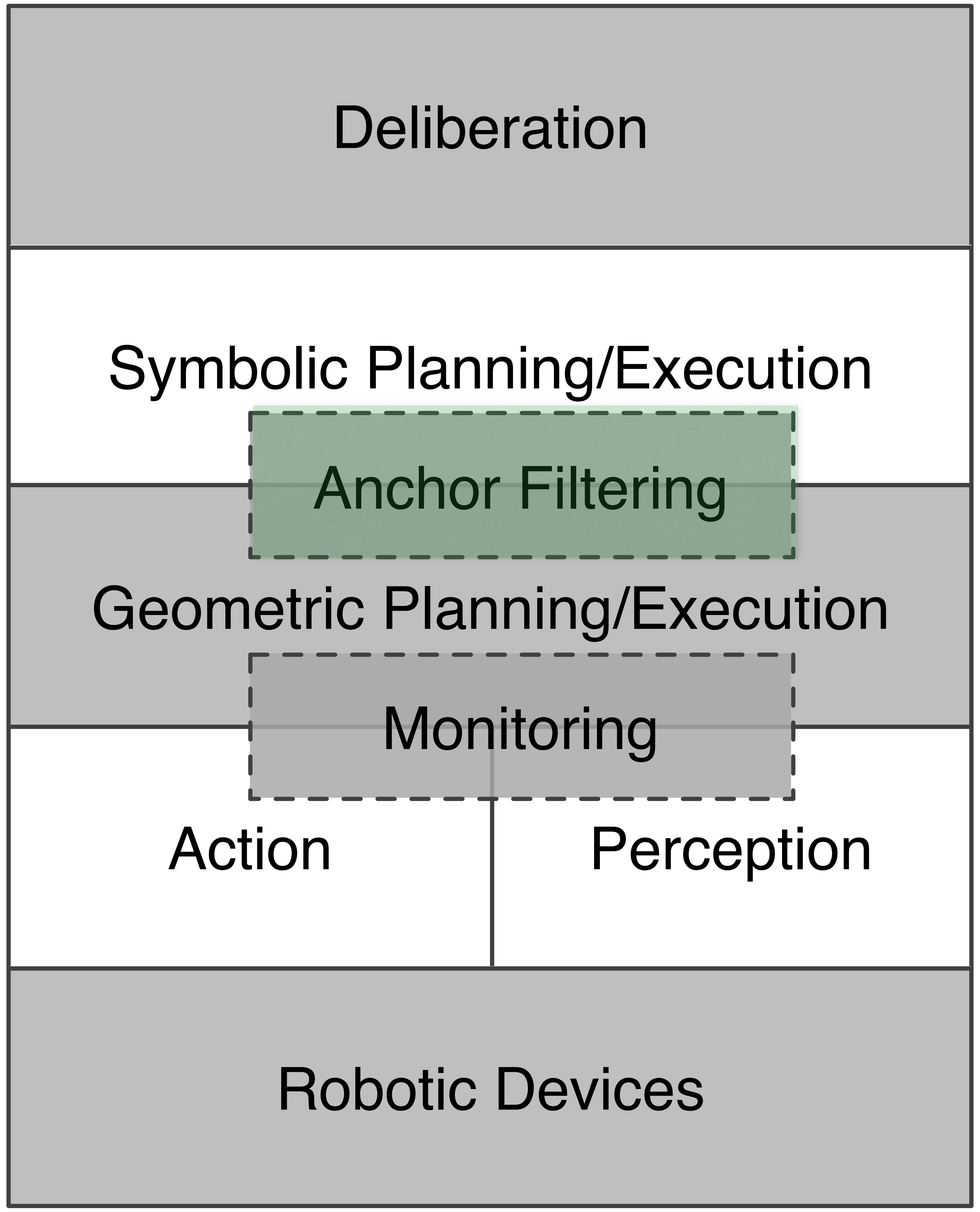
Desiderata - Symbolic Level

- Deliberation: Centered around **declarative goals**
  - Selecting **relevant** goals - a.k.a. desire selection
  - Filtering for **achievable** goals - a.k.a. intention selection
  - Deciding when to give up - a.k.a. **commitment** strategy
- Symbolic Planning and Execution
  - Decomposes goals from deliberation into **discrete tasks** and **actions**
  - Contains an abstracted representation of geometric models



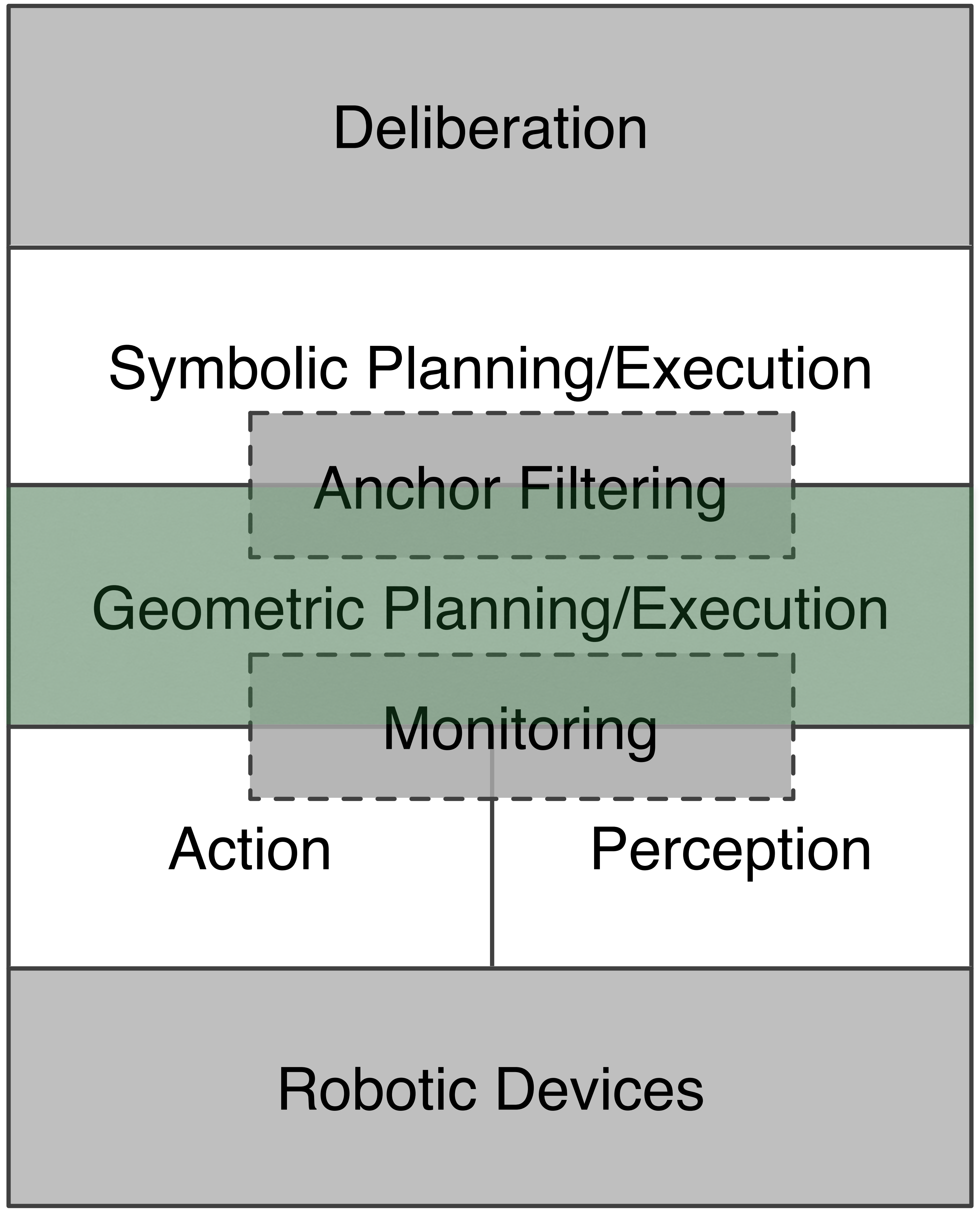
Desiderata - Anchor Filtering

- Infeasible to replicate full 3D models at the symbolic level (even if discretised):
  - Explosion in the number of symbols
  - Inefficiency in logic queries
- Rather, we propose to selectively keep **anchors** between symbolic and geometric level
  - Could be predefined or computed at runtime



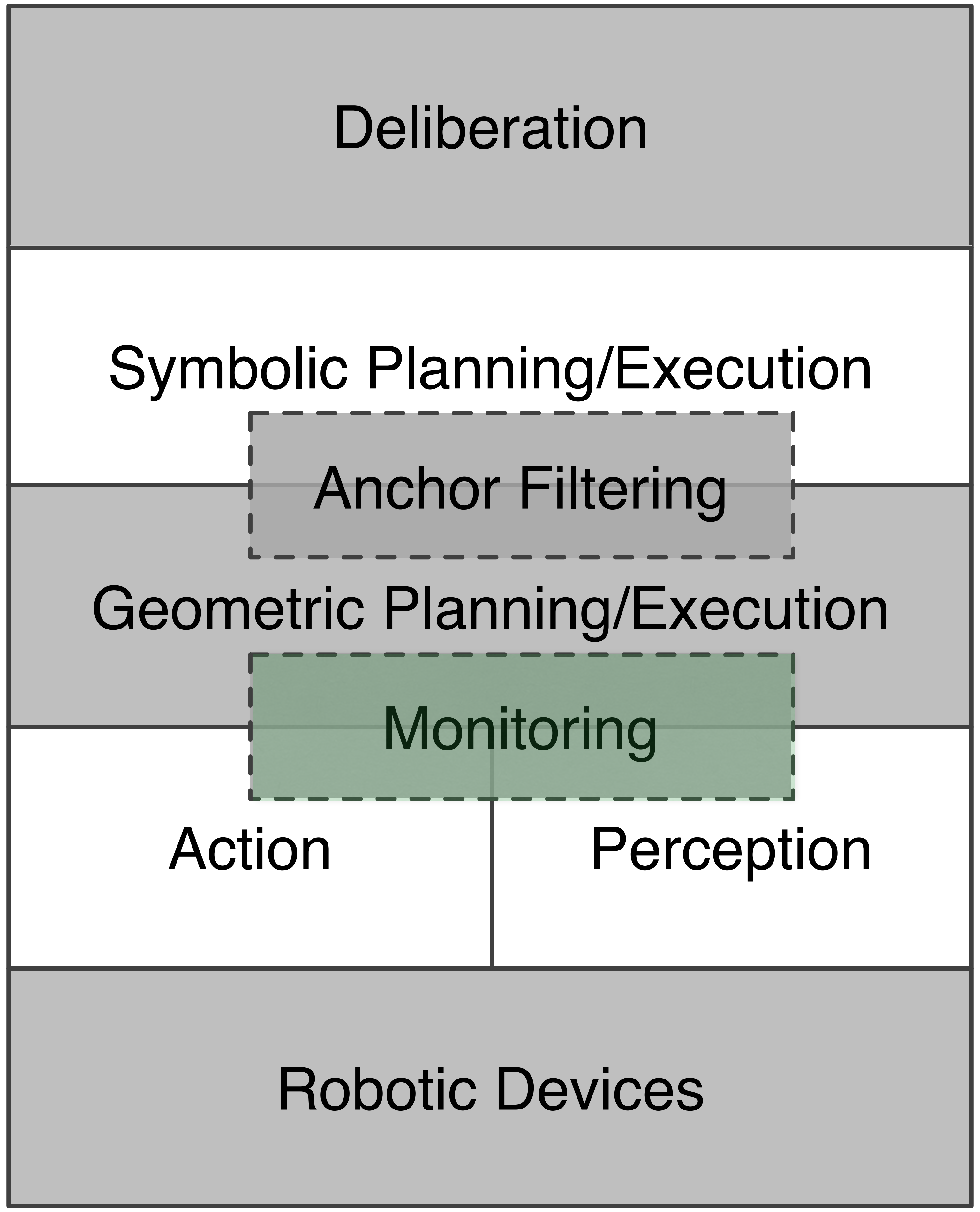
Desiderata - Geometric Planning

- Anchors at the symbolic level need to be evaluated at a finer level of granularity
    - Predicates referring to the 3D world
    - Actions that affect 3D world
- Geometric Planning involves
    - Maintaining a 3D world state
    - Standard 3D motion planning algorithms



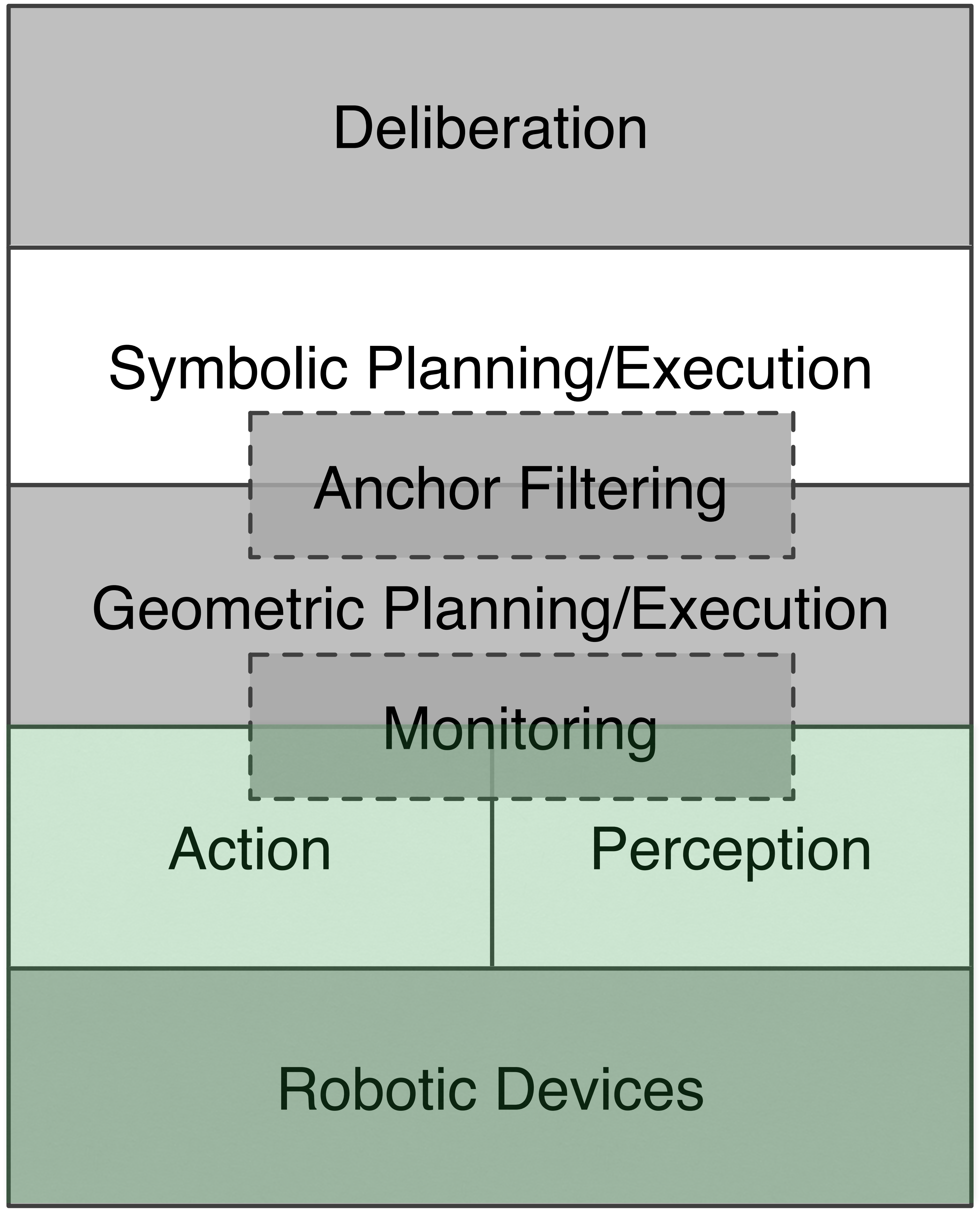
Desiderata - Monitoring

- Continuous monitoring of acting and sensing required for:
  - Critical processes may require realtime reactions — e.g. collision avoidance
  - High-level declarative actions — e.g. moving to location



Desiderata - Action/Perception to Devices

- Action and Perception processing
  e.g. ROS services
  - Raw sensor data processing
  - Complex actuator actions
  - Mixed sensor/actuator processes (SLAM)
- Robotic Devices
  - Translation to specific device implementation



An Instantiation of our Architecture

Instantiation in AgentSpeak(L)

- AgentSpeak(L)
  - Operationalizes the deliberation and symbolic planning layers
  - Many implementations of its semantics, with proven properties
- Key construct: **evaluable/geometric predicates**
  - Main link between Symbolic and Geometric Layers (conceptually filtering)
  - Not linked directly to belief base, but a call to external procedure
  - Call is mediated by a number of functions in filtering

Filtering Layer for AgentSpeak

- Mapping of ground geometric predicates to goal poses — user defined

$$map : C \times P_s \times O_1 \times \ldots \times O_n \rightarrow 2^C$$

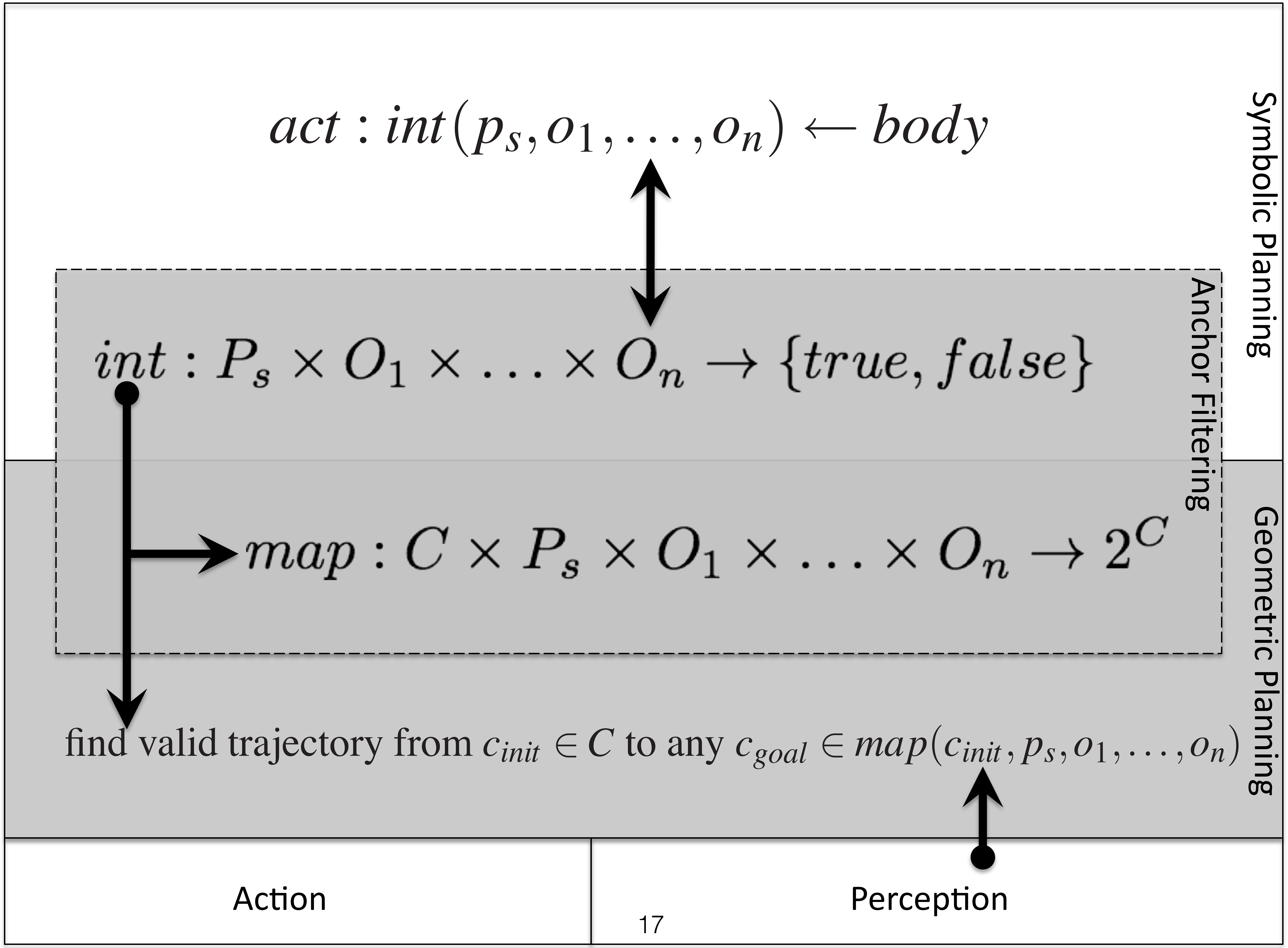
- Intermediary function — mediates calls to geometric planner

$$int : P_s \times O_1 \times \ldots \times O_n \rightarrow \{true, false\}$$

- Intermediary function is called from AgentSpeak preconditions

$$act : int(p_s, o_1, \ldots, o_n) \leftarrow body$$

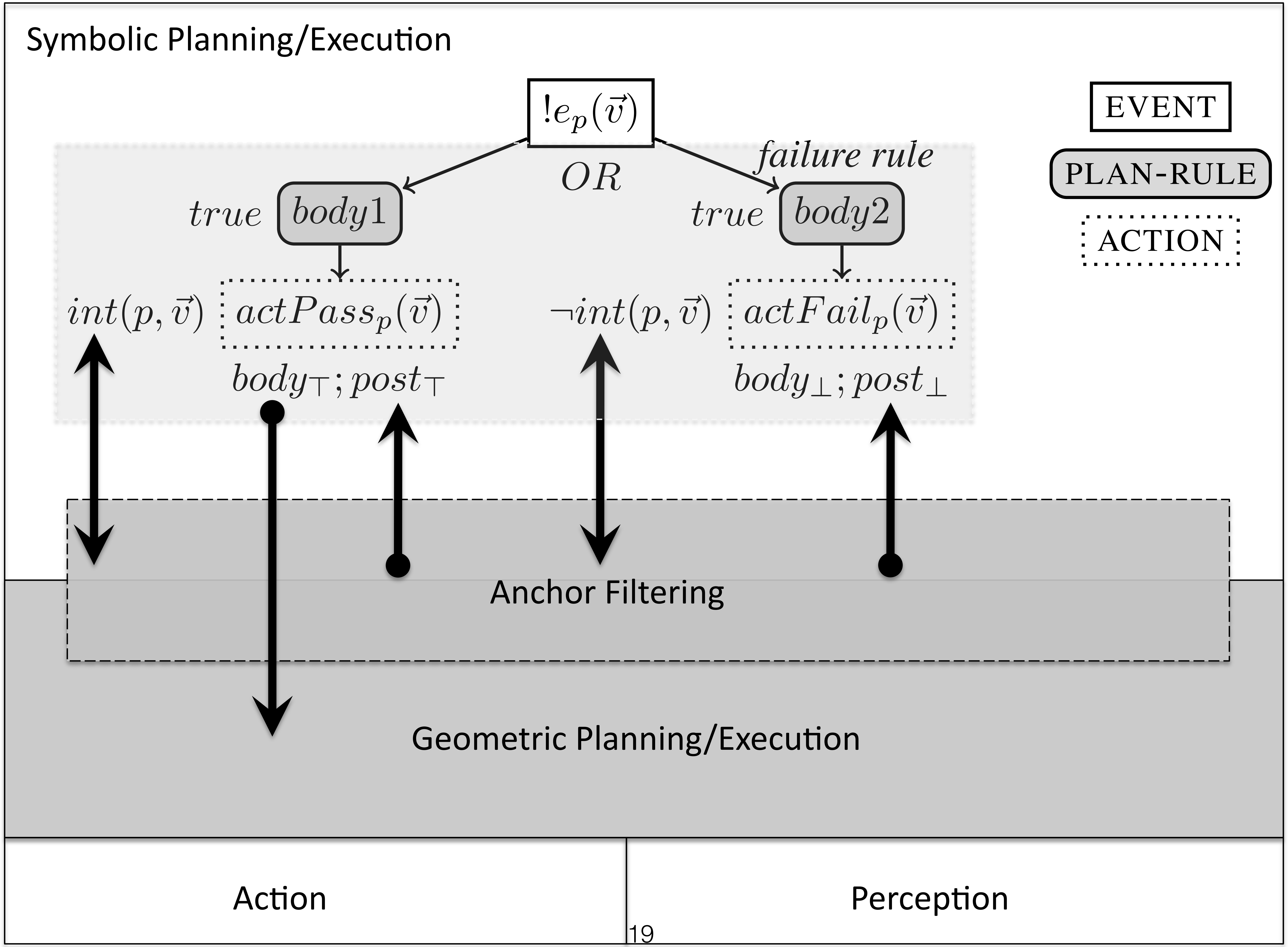
Filtering Layer for AgentSpeak



Connecting with Motion Planning

- Geometric predicates encapsulated in AgentSpeak plan-rules
- Each predicate gets assigned a unique achievement goal
- Achievement goal handled via ***success*** and ***failure*** plan-rules
- Plan-bodies have ***atomic*** actions calling intermediary function, where action-bodies:
    - Execute trajectory found via intermediary function (if any)
    - Add resulting facts, or facts explaining why there's no trajectory

Connecting with Motion Planning



Computing Symbolic Facts

- Action-body obtains domain dependent + independent facts
- Some are computed w.r.t. current pose: e.g. `inside(rob1,room1)`
- New objects—e.g. `cup3`—can be discovered and linked to facts
- Domain independent facts describe non-existence of trajectory
    - `not-reachable(cup1,arm1)` — e.g. `pick(cup1,arm1)` was impossible
    - `obstructsSome(cup3,cup1,arm1)` and `obstructsAll(...)`

Related Work

- Dornhege et al. [2009] — ”Semantic-attachments” and ”Effect applicators”
- Kaelbling et al. [2011,2012] — interleaving planning and execution
- Karlsson et al. [2012], Lagriffoul et al. [2012] and de Silva et al. [2013] — combined symbolic geometric backtracking
- Srivastava et al. [2014] — validating classical plans via geometric trajectories
- Erdem et al. [2011] and Plaku and Hager, [2010] — symbolic planner guides the motion planner toward a collision-free trajectory
- Ingrand and Ghallab [2014] — part of the inspiration for this architecture

Conclusion and Future Work

- Contribution:
  - Summarised functionalities present in existing agent systems and robotic systems
  - Organised them in a tiered architecture
  - Instantiation based on the AgentSpeak(L) language
- Future work: formalise the integration of motion planning in AgentSpeak(L) evaluate an implementation

Acknowledgements

- We thank the helpful discussions with
  - Amit Kumar Pandey (Aldebaran Robotics, Paris)
  - Malik Ghallab (LAAS-CNRS)
  - Alexandre Amory (PUCRS)

Questions?