Extended abstract

Implementing a framework of safe robot planning in Prolog.

Roland Pihlakas
Institute of Technology in University of Tartu
13. august 2008

Note: There are no full papers written yet, nor experiments done. The following abstract describes principles I have devised or borrowed. The experiments will be partially complete during the next few weeks.

The presentation introduces preliminary study for implementing a framework of safe robot planning in Prolog. A principle of safety is introduced which does not depend on explicitly enumerating all possible "negative" states, but at the same time also does not depend on the robot doing only precisely "what it is told to do". The proposed principle of safety is based on implicit avoidance of irreversible actions, except in explicitly permitted cases.

In contrast, usually computer programs do "what they are told to do", but are therefore neither safe nor flexible and autonomous. A possible definition of "robot safety" will be proposed after a few paragraphs.

Previous works about robot safety I am familiar with, proposed the idea of avoiding irreversible actions, see [3], [4], [5]. But these experiments did not contain "positive" goals in them yet. In [2] are additionally explicit "positive" goals in use. Ideas of "optional" goals and goal-specific permissions are also introduced. In [1] one can find a possible implementation of modal logic in Prolog.
All of the previously mentioned studies depend on explicitly enumerated "negative" goals, or on explicitly enumerated irreversibilities which should be avoided.

I propose that avoiding irreversibilities should be implicit. Most of the possible "negative" consequences should not be explicitly enumerated in the robot's configuration, because it is very easy to miss something in such kind of configuration, especially when situation may change in the future.

It is also important that the plans are constructed by the robot and not all actions in the plan are explicitly listed in the user's request. The robot should be able to learn and to automatically decide all necessary sub-~~steps~~ in the plan towards fulfilling explicitly given goals, avoiding at the same time actions with "negative" consequences.

The term "safety" means in this study avoiding unwanted or unforeseen consequences of plans**. At the same time, skipping unforeseen / unrequested positive consequences is acceptable.** Therefore in order to be practical, a safe robot needs to achieve only these goals which are explicitly requested (or are **acceptable** sub-goals of the request) and has to avoid any results that are of unknown value.

Avoiding unwanted results of a plan means here that only these consequences of actions should be avoided, which cannot be reversed, or "cleaned up" during the execution of plan. Therefore reversible actions are by default not considered as "negative", unless they have some "cost" associated with them.

From that idea emerges the principle of implicitly avoiding all irreversible actions, except these ~~actions~~ for which explicit permission has been given. A safe robot should use only such sub-goals that will cause predictable and permitted changes in the environment.
It is also advisable that a safe robot will prevent only own-caused mistakes from happening. It should not try to prevent others from doing mistakes, because that entails complex issues with priorities and rights.
Thirdly, a safe robot acts only towards these goals or changes (in the environment) that it has been ordered to, or which are necessary sub-goals for achieving some given order. That means that the robot is not in any other way restless either. Therefore it is always possible to assign human responsibility to everything the robot does.

Because avoiding non-permitted irreversibilities is of highest priority, the robot does not follow orders when it does not have enough rights (permissions) for executing the full plan. The permissions must be specified explicitly in the robot's configuration.

The robot may also optionally have in its configuration file some enumerated states that are explicitly listed as having known negative value. An example of this use is a control system, which balances something.

In the proposed implementation, the goals to avoid irreversibilities in any modality have always higher priority than any explicitly given "positive" goal, unless this certain modality is explicitly listed as only "optional" for avoiding irreversibilities. An example of optional avoidances is "avoid littering crap around yourself, when you are unable to clean it up later!"

The term "modality" means here: a certain aspect of the world, which can be measured or computed.

There are also optional "positive" goals or recommendations, like "clean up the temporary reversible mess when you have spare time left!", or "prefer method A to method B!".
The term "optional" means here that "don't abort rest of the plan if unable to follow this recommendation!"

The permissions that are given, are necessarily context-specific, depending on the robot's competence area.

A special kind of context is a “sandbox” where making “mistakes” by causing irreversible changes in the environment is permitted for training purposes. Later the human user, based on the robot’s competence in its future work area, gives some specific permissions to the robot. This way the responsibility will be on human’s shoulders.
The permissions can be used for achieving explicitly given goals, but do not have to be used.

The previously mentioned ideas can be summarised with the following priority-list:

1. Explicit permissions of three kinds:
   a) permissions for actions;
   b) permissions for changes in certain modalities;
   c) permissions for achieving certain states.
2. Implicit avoidances of irreversible actions. That is - avoidances of implicitly non-permitted changes in modalities.
3. Explicit “negative” goals - avoidances (optional). **TODO: priority over positive goals, is this done?**
4. Explicit “positive” goals, usually in the form of request to achieve certain states.
5. “Optional” goals and avoidances.

Explanation about using modalities: because same action can have different consequences in different situations / contexts, usually the permissions are given for changes in certain modalities, instead of enumerating all the actions that could cause these changes.

I call the proposed principle of safety a “passive safety”. “Passivity” means here that the robot distinguishes clearly between the orders that were given and the sub-goals it has set to itself. The consequence of this distinction is that the robot will not try to make things “better” if not ordered to do so; and will not agree to do many actions, even if these actions are possible sub-goals of a given task. The most important part about the passivity is that refusing to do actions is the “the first law” and following the orders is only “the second law”.
Another important but optional principle of “passive” safety is the following: the robot should rather avoid only own mistakes. This is opposed to proactive avoidance of changes caused by external agents or natural causes.

In case of uncertain or adverse environments one can use Minimax algorithm to find plans that can cause minimal costs and possibly maximal gain.

Using this system of priorities and permissions one usually should not enumerate explicitly the “negative” states. I already mentioned that relying only on the enumeration of risks is by itself risky **(TODO: where?).** There is an additional reason for not enumerating “negative” states**. One should be wary of the issue that many states would be labelled as “negative” only because we do not want the robot causing them**. But at the same time we do find it okay or even important to cause these states through human hand. If we configured related modalities by enumerating specific “negative” states in the robot’s configuration, we could find ourselves in the situation where the robot tries to reverse even these states we manually caused (and do not want them to be reversed). Therefore, it is better to configure the robot so that it simply avoids own-caused changes of certain kind (that is, changes in certain modalities), irrespective of the direction of change.
In contrast, goals usually describe specific states to be achieved.

As mentioned previously, the context of robot's permissions and goals is important. Every entry in the robot's configuration may include the following fields:

- type of entry: permission or goal;
- type of change: action, state or modality;
- place;
- time (or time limits for achieving a goal);
- in case of permissions, the specific goals for which this permission is enabled;
- priority;
- weight / cost per ~~size~~ of change, or per duration (also: time-windows for reversing certain actions);
- weights / costs for probabilities (separate issue from the above mentioned weights, because some improbable but high-impact situations should still be avoided at all costs);
- whether changes from external causes should also be balanced / reversed by the robot;
- whether the permission is given for a permanent change or only for a temporary change that must be reversed before the completion of a plan;
- for goals: whether it is okay to achieve the plan partially, or shouldn't the partially accomplishable plan be executed at all.

References

[1] J. Fox, S. Das, *Safe and sound: Artificial Intelligence in Hazardous Applications*. London: AAAI Press / The MIT Press, 2000

[2] D. Weld, O. Etzioni, The First Law of Robotics (a call to arms), *AAAI-94, 1994*. URL: http://www.cs.washington.edu/homes/etzioni/papers/first-law-aaai94.pdf

[3] A. Eppendahl, M. Kruusmaa., "Obstacle Avoidance as a Consequence of Suppressing Irreversible Actions", *Proceedings of EpiRob*, 2006. URL: http://homepage.mac.com/a.eppendahl/work/papers/Eppendahl-obsacs.pdf

[4] A. Eppendahl, M. Kruusmaa, Y. Gavshin, "Don't Do Things You Can't Undo: Reversibility Models for Generating Safe Behaviours", 2007. URL: http://homepage.mac.com/a.eppendahl/work/papers/Eppendahl-dondty.pdf

[5] J. Gavšin, "Using the Concept of Reversibility to Develop Safe Behaviours in Robotics" (dissertation). Tartu: Tartu Ülikooli Arvutiteaduse instituut, 2007. URL: http://hdl.handle.net/10062/1990