Robot Control Interface - Feature #944
Provide pose interpolation

03/13/2012 04:17 PM - C. Emmerich

Status:	Feedback	**Start date:**	03/13/2012
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	0%
Category:		**Estimated time:**	0.00 hour
Target version:	rci0.5		

Description

Here is some code snippet that can be useful for that:

```
vector<Pose> interpolatePoses(const Pose& poseStart, const Pose& poseEnd, unsigned int numSamplingPoints) {
  vector<PosePtr> poses;

  if (numSamplingPoints == 0) {
    poses.push_back(poseStart);
    poses.push_back(poseEnd);
    return poses;
  }

  ++numSamplingPoints;
  double ratio = 1.0/numSamplingPoints;
  RealVector poseStartValues = poseStart.asDoubleVector();
  RealVector poseEndValues = poseEnd.asDoubleVector();
  for (unsigned int p = 0; p <=numSamplingPoints; ++p) {
    RealVector poseValues = poseStartValues * (1 - p*ratio) + poseEndValues * (p*ratio);
    poses.push_back(Pose(poseValues));
  }

  return poses;
}
```

It creates a vector of *numSamplingPoints + 2* poses (= *start pose + interpolation poses + end pose*).

History

#1 - 08/12/2013 02:10 PM - Anonymous

- Status changed from New to Feedback

- Assignee changed from Anonymous to C. Emmerich

- Target version set to rci0.4

Christioan, would you want to include this into the RCI's Pose API?

#2 - 08/12/2013 04:41 PM - C. Emmerich

What do you mean by "Pose API"?

If you mean something like

```
class Pose {
  public:
    Pose interpolate(const Pose p, unsigned int numInterpPoints) {
      // interpolate between this and p
    }

    ...

}
```

that doesnt make any sense in my opinion.

But it would be nice to have such method in a central place when dealing with Poses, and why not in the Poses class, e.g. by means of a static function?

```
class Pose {

  public:
    static std::vector<Pose> interpolate(const Pose& pose1, const Pose& pose2, unsigned int numInterpPoints) {
      // interpolate between pose1 and pose2 and return vector of Poses
    }

  ...

}
```

On the other hand, with including such a method in the Poses class (which is a rather rudimentary functionality providing only **linear** interpolation in translatory space and whatever should happen in rotatory space) the question arises why not including more such feasibility features like e.g. spline interpolation, different rotatory interpolations, or even different things like e.g. drawing random Poses around a mean Pose etc., which obviously would clutter the Poses class...

Maybe it would be better to provide something like a "utils package/class/orwhatever" whithin the RCI lib...?

#3 - 08/12/2013 04:42 PM - C. Emmerich

- Assignee changed from C. Emmerich to Anonymous

#4 - 09/12/2013 04:25 PM - Anonymous

- Target version changed from rci0.4 to rci0.5