AltTF: Alternative lock-free TF tree

荻原湧志, 川島英之

December 18, 2021

概要

TF は ROS でよく使われるパッケージであり、これは有向木で座標変換を管理する。この木構造には Giant lock によるパフォーマンス低下の問題、線形補間による一貫性の欠落の問題があった。そこで、我々はトランザクション技術における 2PL を導入することにより、この問題を解決した。

1　序論

ROS はロボットソフトウェア用のミドルウェアソフトプラットフォームであり、近年多くの研究用ロボットで使われている

グローバル座標系の中でのロボットの位置、ロボットのローカル座標系の中でのセンサーの位置、センサーのローカル座標系の中での物体の位置、という風に別々に管理している

この時、グローバル座標系での物体の位置を計算するにはグローバル座標系からロボットの位置の計算 * センサーの位置 * 物体の位置を計算すれば計算できる

TF ではこのような座標系同士の変換を次の図のように有向森構造で管理できる。座標系の原点を frame と呼び、このフレームを node、座標系同士の位置関係を並行移動成分・回転成分で edge で表した木構造で管理する木構造で管理することにより、先程のようなグローバル座標系での物体 A の位置は、二つの frame 間のパスを辿ることで計算できるロボットの座標変換の情報は全てこの TF モジュールで管理される

TF モジュールは ROS の多くのパッケージで使われているが、これには以下のような問題点がある

Giant lock により、木構造が大きくなり多くのスレッドがアクセスするケースでは遅くなる

(setTransform の non atomicity は実はまだ問題にはならない)

提供されているインターフェイスでは「最新」のデータを取るというメソッドでも実際には該当するパスのデータが完全に準備された状態の時間のデータを返している。

また暗黙的に線形補完が行われるため、データの一貫性がなくなる可能性がある

1.1　貢献

そこで、データベースのトランザクション機構における並行性制御のプロトコルの一種である 2PL を導入した。

その結果、224 コアのマシンではこの程度の性能差が出た

また、データ一貫性を保証できるような setTransforms, lookupLatestTransform というインターフェースを導入した。

1.2　論文構成

本論文の構成は次の通りである。2 章では既存の TF 森の構造とその問題点について述べる。3 章では提案手法である TF 森の再粒度ロックの導入とデータ一貫性のためのインターフェイスの提供について述べる。4 章では提案手法の評価結果を述べる。5 章では本研究の結論を述べる。6 章では今後の課題について述べる。

2　既存の TF 森の構造とその問題点

TF 森の実態は tf2 パッケージ中にある BufferCore クラス [4] である。

2.1　構造

まずは TF 森とタイムテーブルから？

各座標系同士の回転移動、並行移動で表現できる位置関係は TF 森で表現される

例えば図のようなマップ座標系、ロボット座標系、カメラ座標系、物体の座標系は図のような木構造に対応する。

また、図のフレーム A, B のように他の木とは分離された木が存在してもよい。このため、TF 森構造となる。

さて、座標変換の情報は変わらない場合と時刻とともに変わる場合がある

さて、座標変換の情報は主にセンサーからデータが送られてくるたびに計算され、TF 森に登録される。このため各フレーム間の座標変換の情報はセンサーの更新頻度に依存し、それぞれ異なるタイミングで更新される。例えば、マップ座標系からロボット座標系の座標変換を計算するプログラムは LiDAR のデータ更新頻度に合わせて自己位置を計算し、マップ座標系からロボット座標系の座標変換を登録する。カメラ座標系から物体の座標系の座標変換を計算するプログラムはビデオカメラのデータ更新頻度に合わせて物体の位置を計算し、カメラ座標系から物体の座標系の座標変換を登録する。カメラの更新頻度と LiDAR の更新頻度が異なる場合、マップ座標系からロボット座標系の座標変換が登録されるタイミング、カメラ座標系から物体の座標系の座標変換が登録されるタイミングにズレが生じる。マップ座標系から物体の座標系を計算する際にデータ同期の問題が生じてしまう。これに対処するため、まず TF 森は各フレーム間の座標変換情報を過去一定期間保存する。これは図のように表現できる

図は各フレーム間の座標変換情報が提供される時刻を表す。横軸が時間軸を表し、左側が過去、右側が最新の時刻を表す。

灰色線は対応する時刻の座標変換情報が登録されたことを表す。図の点線の位置での map から apple への座標変換を計算する際、各フレーム間の座標変換情報が必要になる。robot から camera への座標変換情報は常にある物として扱われるが、map から robot、camera から apple の座標変換データは該当する時刻では登録されていない。ここで、TF は前後のデータから線形補間を行うことにより座標変換データを計算する。

文字列である frame は TF 森内部では整数型の id で管理している frame から frame id を検索するテーブル、及び frame id から frame を検索するテーブルが存在する

座標変換は TransformStorage で表現され、これは以下で構成される回転成分を表す rotation。これは Quaternion で表現される並行移動成分を表す translation。これは Vector3 で表現される座標変換の時刻を表す stamp。これは ros::Time 型で表現される座標変換の親フレームの frame id を表す frame_id。これは整数型で表現される座標変換の子フレームの frame id を表す child_frame_id。これは整数型で表現される

TimeCache はある子フレームにおける親フレームへの座標変換を時系列的に管理する。これは TransformStorage の deque で構成され、ユーザーが座標変換情報を登録する際には先頭に push し、保存していた座標変換の時刻が 10 秒以上過去のものとなれば deque から追い出す。

StaticCache は先程の例の robot-¿sensor 間の座標変換のように不変なある子フレームにおける親フレームへの座標変換情報を管理する。

TimeCache と StaticCache は TF 森における子フレームから親フレームへのエッジ及び子フレームのノードを表す。TimeCacheInterface は TimeCache と StaticCache の親クラスであり、各フレーム id に対応する TimeCacheInterface のポインタを管理する配列が各フレームに対応する TimeCache と StaticCache を管理する。

2.2 検索

TF には lookupTransform というメソッドがあり、

2.3 更新、挿入

2.4 問題点

3 提案手法

提案手法と snapshot latest の概念の導入について時系列データにはいかなる変更も加えられないため serializability を導入する必要はないまた、最新のデータを見るには serializability が必要

4 評価

実験を行う

5 結論

データの正確性は必須

6 今後の課題

ここでは取り上げなかった TF 木の問題点として、部分的に

References

[1] Simon J. Julier and Jeffrey K. Uhlmann ”Building a million beacon map”, Proc. SPIE 4571, Sensor Fusion and Decentralized Control in Robotic Systems IV, (4 October 2001)

[2] M. Quigley, K. Conley, B. P. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, “Ros: an open-source robot operating system,” in ICRA Workshop on Open Source Software, 2009.

[3] T. Foote, ”tf: The transform library,” 2013 IEEE Conference on Technologies for Practical Robot Applications (TePRA), 2013, pp. 1-6, doi: 10.1109/TePRA.2013.6556373.

[4] ”BufferCore.h”, https://github.com/ros/geometry2/blob/noeti devel/tf2/include/tf2/buffer_core.h