The Multi-State Constraint Kalman Filter

Or, how we learned to stop worrying and love the null space

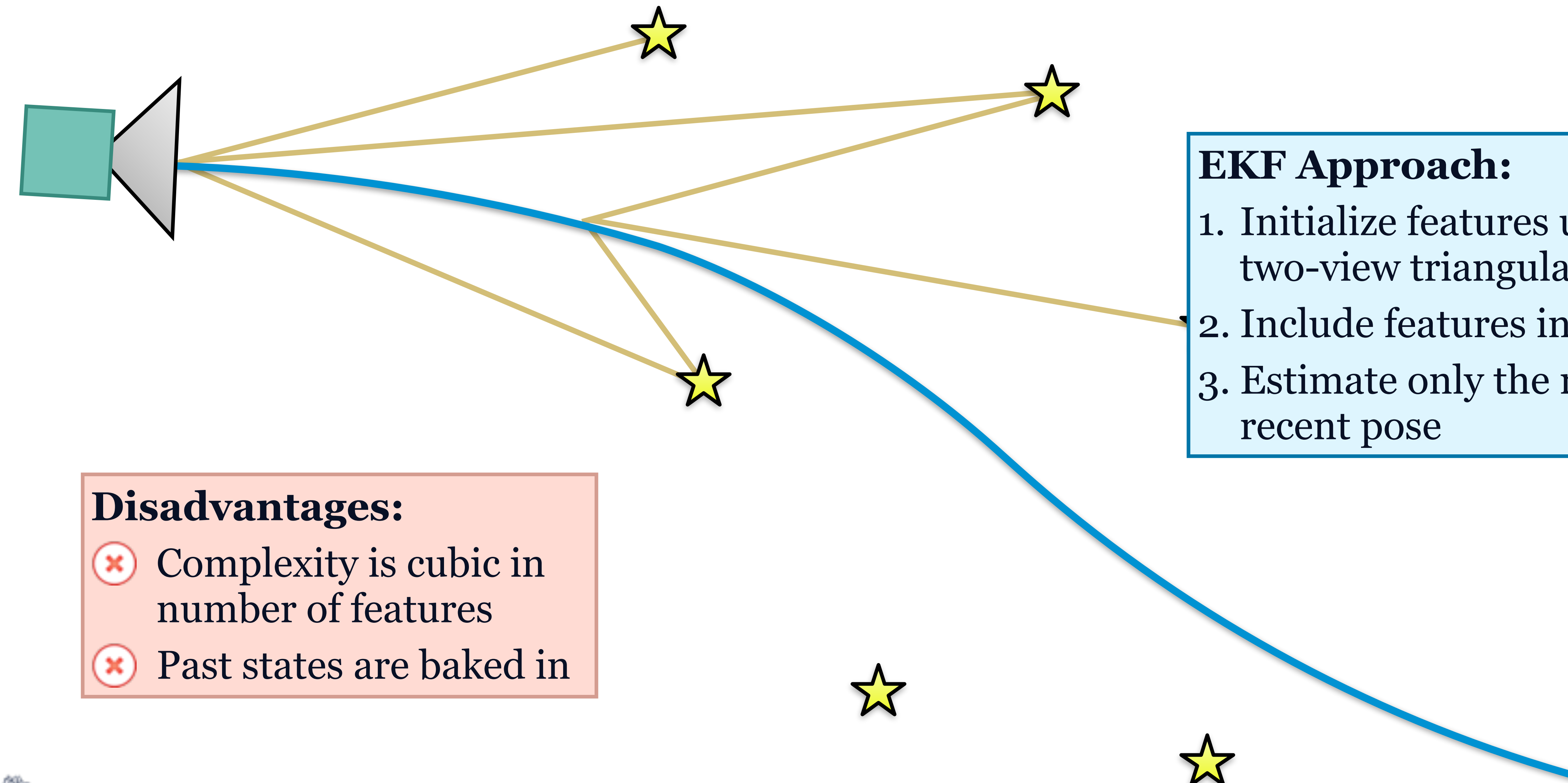
Valentin Peretroukhin and Lee Clement

AER1513 Course Project

Institute for Aerospace Studies
UNIVERSITY OF TORONTO

Problem: Mapless Navigation

Goal: Use an IMU with a monocular camera to estimate motion *without a map*.



EKF Approach:

1. Initialize features using two-view triangulation
2. Include features in state
3. Estimate only the most recent pose

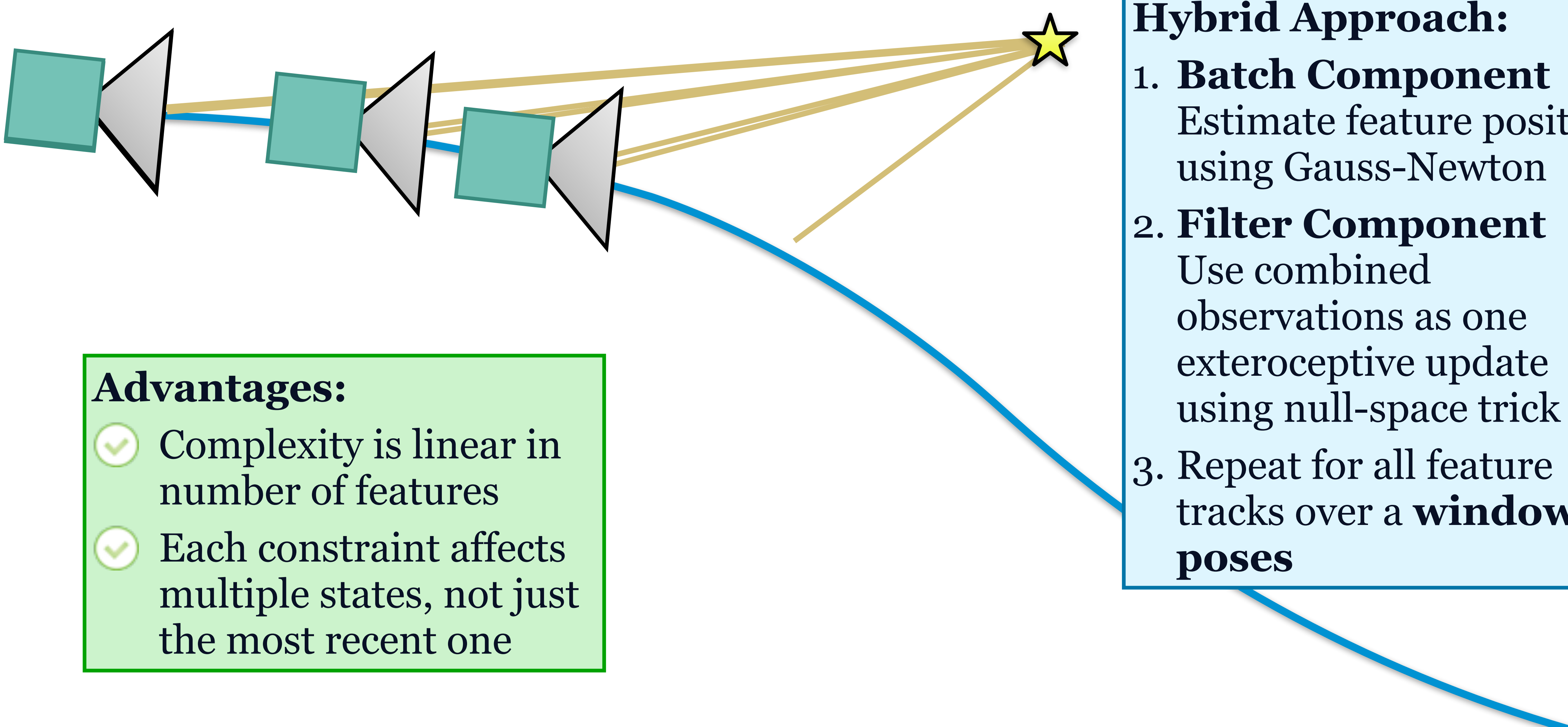
Disadvantages:

- ⊗ Complexity is cubic in number of features
- ⊗ Past states are baked in

Institute for Aerospace Studies
UNIVERSITY OF TORONTO

Algorithm: Multi-State Constraint Kalman Filter

MSCKF dead-reckons the vehicle state using interoceptive (IMU) measurements, just like the EKF, but treats exteroceptive (camera) measurements differently.



Hybrid Approach:

1. **Batch Component**
   Estimate feature position using Gauss-Newton
2. **Filter Component**
   Use combined observations as one exteroceptive update using null-space trick
3. Repeat for all feature tracks over a **window of poses**

Advantages:

- Complexity is linear in number of features
- Each constraint affects multiple states, not just the most recent one

MSCKF: Null Space Trick

What we want: $\mathbf{e}_{\text{ext}} = \mathbf{G}_{\mathbf{x}}\delta\mathbf{x} + \text{noise}$

What we've got: $\mathbf{e}_{\text{ext}} = \mathbf{G}_{\mathbf{x}}\delta\mathbf{x} + \mathbf{G}_{\mathbf{p}_f}\delta\mathbf{p}_f + \text{noise}$

$$\mathbf{N} := \text{Null}\left(\mathbf{G}_{\mathbf{p}_f}\right)$$

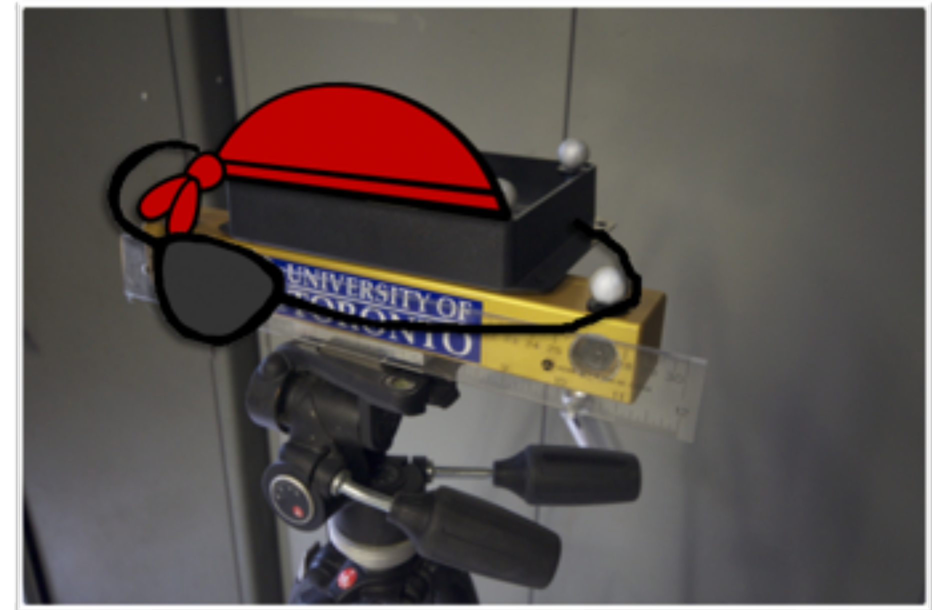
$$\mathbf{0} \quad \because \quad \left(\mathbf{N}^T\mathbf{G}_{\mathbf{p}_f} = \mathbf{0}\right)$$

$$\mathbf{N}^T\mathbf{e}_{\text{ext}} = \mathbf{N}^T\left(\mathbf{G}_{\mathbf{x}}\delta\mathbf{x} + \mathbf{G}_{\mathbf{p}_f}\delta\mathbf{p}_f + \text{noise}\right)$$

$$\mathbf{e}'_{\text{ext}} := \mathbf{G}'_{\mathbf{x}}\delta\mathbf{x} + \text{noise}'$$

Dataset: Starry Night (Assignment 3)

- ☑ Perfect data association
- ☑ Ground truth for landmark positions
- ☑ Pre-integrated IMU measurements



Planned Analysis: MSCKF vs. Sliding Window

We will **investigate** MSCKF parameters:

1. Feature track length
2. Maximum window size

We will **compare**:

MSCKF *vs.* Sliding Window
Batch Estimation *without a map*

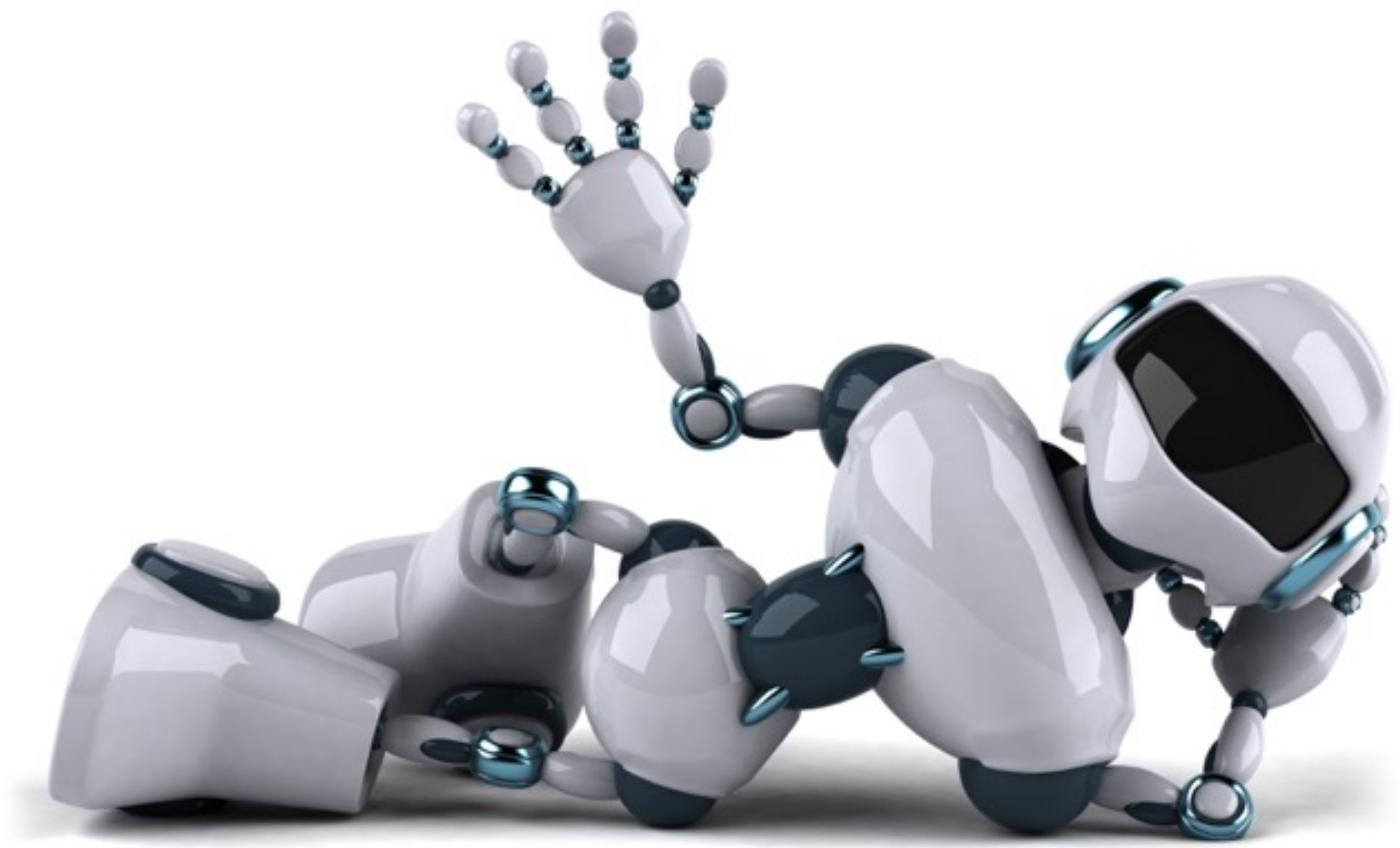
Multi-**S**treet **C**omparison **K**ickboxing **F**ight

Proof of Progress

- MSCKF.m
- propagateImuState.m
- propagateMsckfCova...
- augmentState.m
- updateState.m
- calcF.m
- calcG.m
- calcJ.m
- calcHoj.m
- calcTH.m
- calcGNPosEst.m
- calcResidual.m
- getGoodGrade.m

```
%==========================STATE PROPAGATION========================%

%Propagate state and covariance
msckfState = propagateMsckfCovar(msckfState, measurements{state_k}, noiseParams);

%Add camera pose to msckfState
msckfState = augmentState(msckfState, camera);


%==========================FEATURE TRACKING========================%
% Add observations to the feature tracks, or initialize a new one
% If an observation is -1, add the track to featureTracksToResidualize
featureTracksToResidualize = {};
for featureId = 1:20
    meas_k = measurements{state_k}.y(:, featureId);
    if ismember(featureId, trackedFeatureIds)
        if meas_k(1,1) == -1
            %Add to residualize queue and remove from the original
            %struct
            featureTracksToResidualize{end+1} = featureTracks{trackedFeatureIds == featureId};
            featureTracks = featureTracks(trackedFeatureIds ~= featureId);
            trackedFeatureIds(trackedFeatureIds == featureId) = [];
        else
            %Append observation and increase k2
            featureTracks{trackedFeatureIds == featureId}.observations(:, end+1) = meas_k;
            featureTracks{trackedFeatureIds == featureId}.k2 = featureTracks{trackedFeatureIds == featureId}.k2 + 1;
        end
    else
        %Track new feature
        track.featureId = featureId;
        track.observations = meas_k;
        track.k1 = state_k;
        track.k2 = state_k;
        featureTracks{end+1} = track;
        trackedFeatureIds(end+1) = featureId;
    end
end

%==========================FEATURE RESIDUAL CORRECTIONS========================%

featuresToResidualize = []; %1xN matrix of feature ids (this is just the column of y_k_j)
```

Thanks!



Institute for Aerospace Studies
UNIVERSITY OF TORONTO

Image credit: galleryhip.com

Extra Slides

Algorithm: Multi-State Constraint Kalman Filter

Idea: Use a hybrid batch/recursive filter to incorporate all observations of a feature *without storing it in the state vector*.

Batch component: Track each feature until it goes out of view, then compute its position from *all available measurements* using Gauss-Newton optimization.

Recursive component: Use landmark position and all of its measurements (with null space trick) *to constrain motion*.

Advantages over plain EKF:

- **Sliding window of poses** allows each constraint to affect multiples states
- **Computational complexity is linear** in number of landmarks instead of cubic for plain EKF SLAM.