RVSS 2026

Part I: Modelling the world

Prof Tom Drummond

Building models of the world

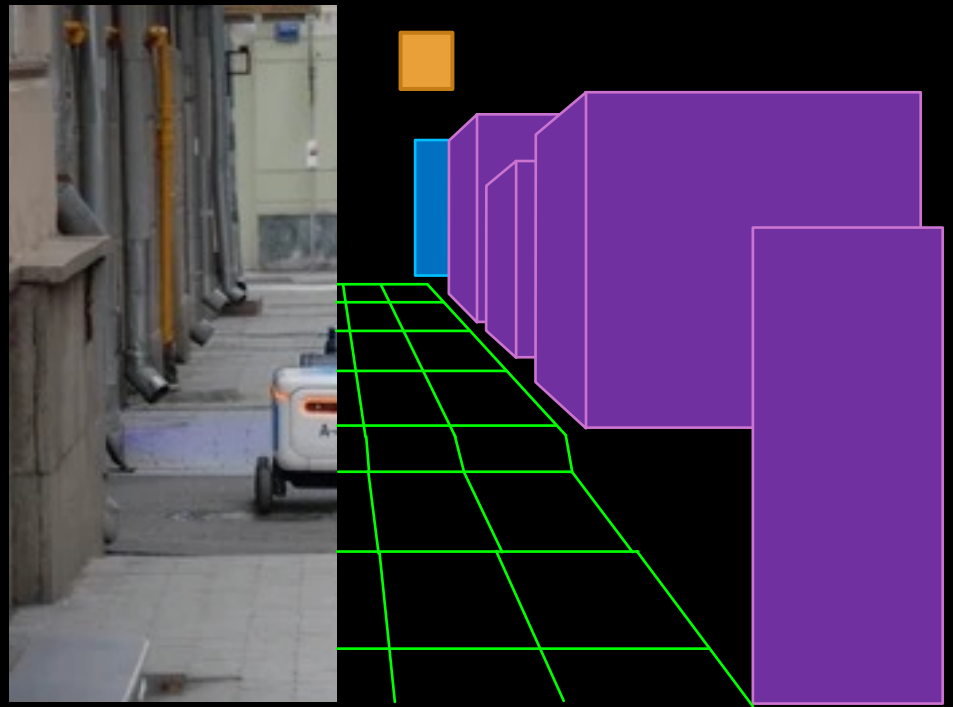
A model is some kind of digital representation of the world around the robot

Building models of the world

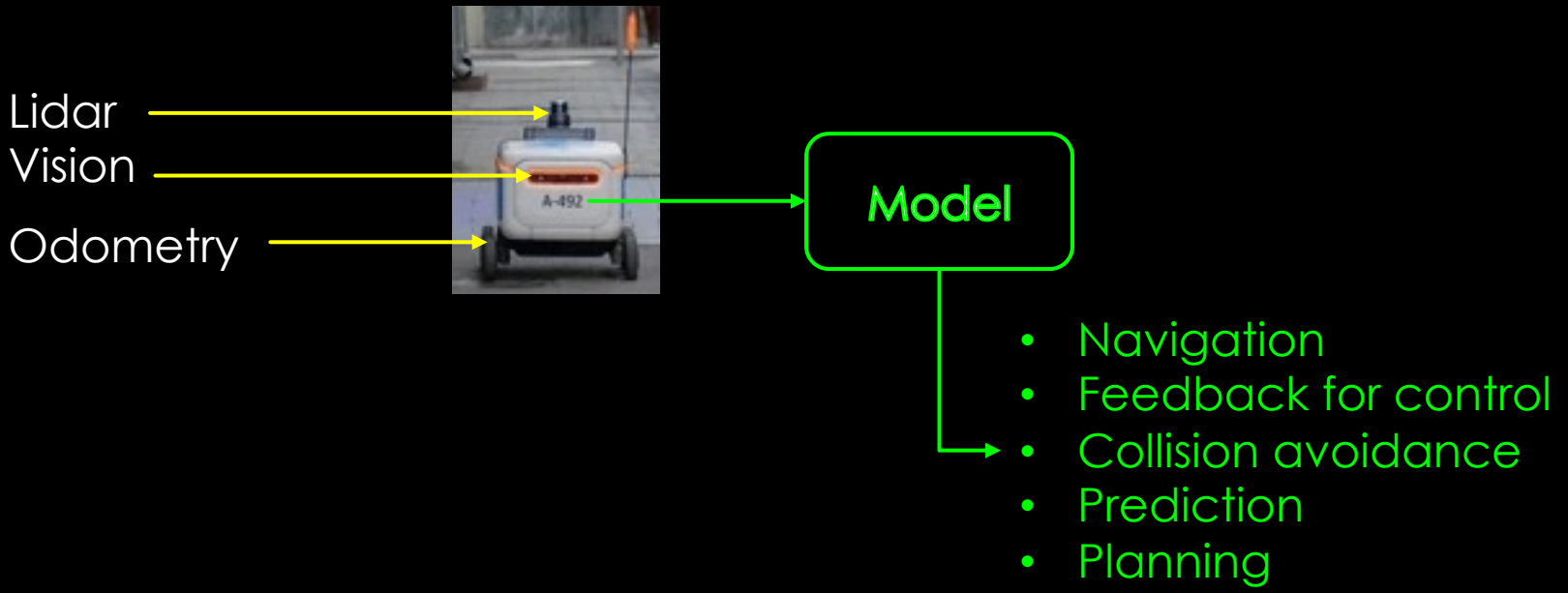
A model is some kind of digital representation of the world around the robot

All models are wrong…

Why build models?



BUILDING MODELS OF THE WORLD



VISION = GRAPHICS^{-1}

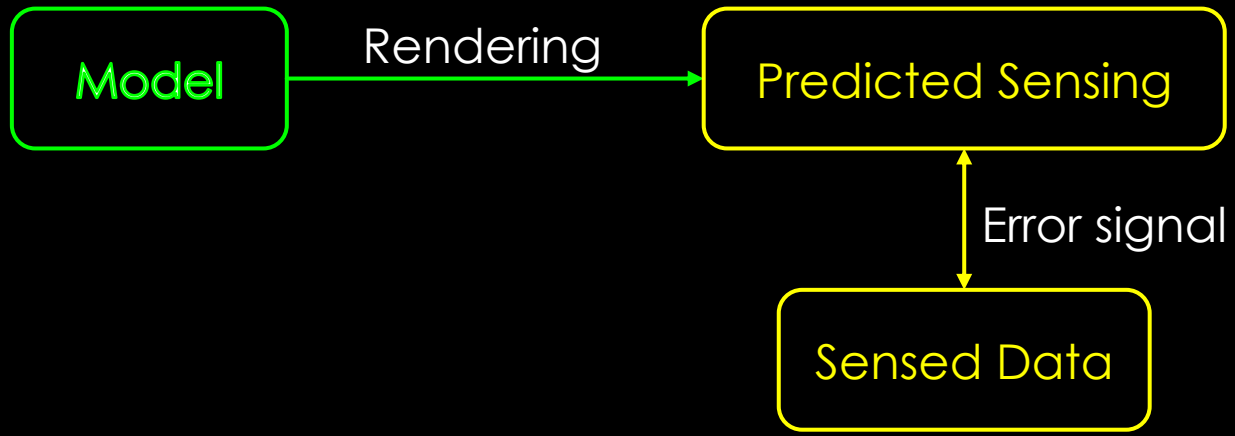
Modelling takes sensed data as input to create a model

VISION = GRAPHICS^{-1}

Often build models by inverting the problem

Use current model to predict what we sense

Use the error to refine the model



Kinds of models

Models may contain:

- Geometry (where stuff is, usually 2D or 3D)
- Semantics (what stuff is)

Model Geometry can be:

- Sparse (just enough landmarks to compute robot motion)
- Dense (mostly complete representation of scene structure)

Model Semantics can represent:

- Stuff (continuous material: road surfaces, grass, rock, etc.)
- Things (discrete objects: cars, pedestrians, road furniture, etc.)

Kinds of models

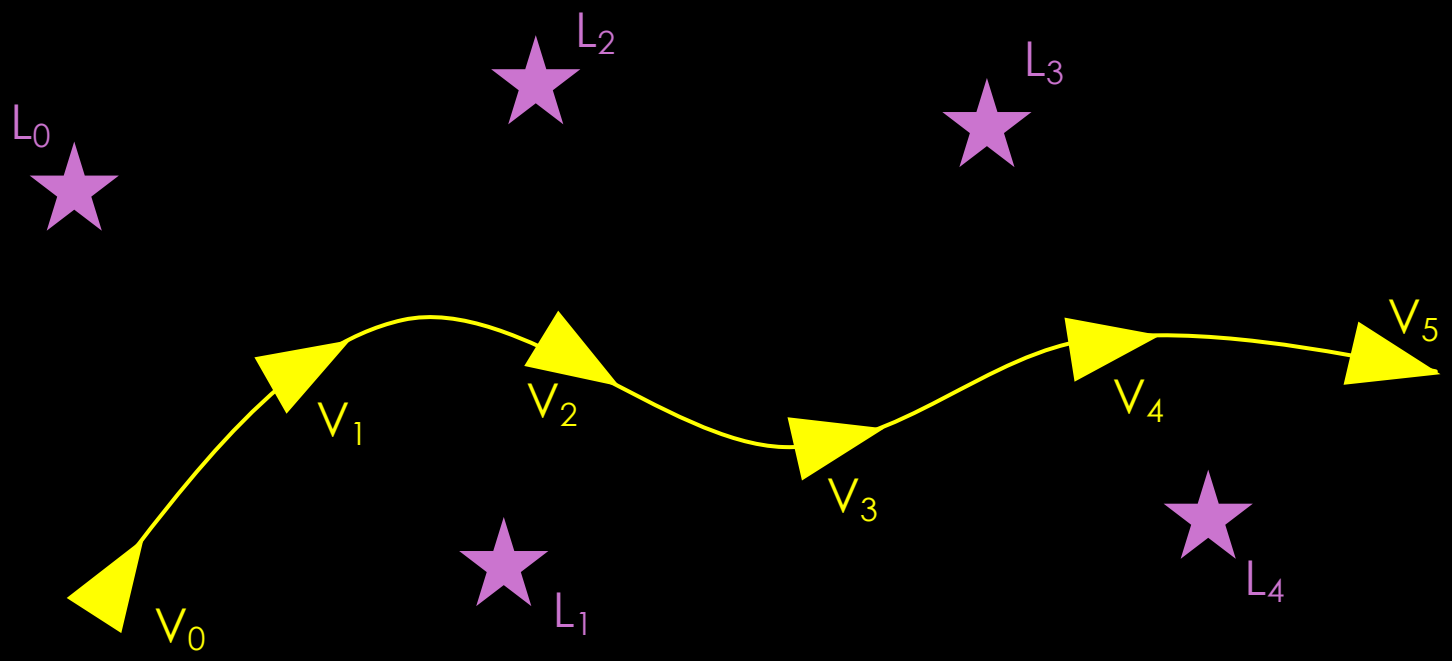
Models can be used to:

- Remember the *past* (e.g. build a map so we can navigate home)
- Represent the *present* (e.g. compute position for motion control)
- Predict the *future* (e.g. plan motion to avoid collisions with moving objects)

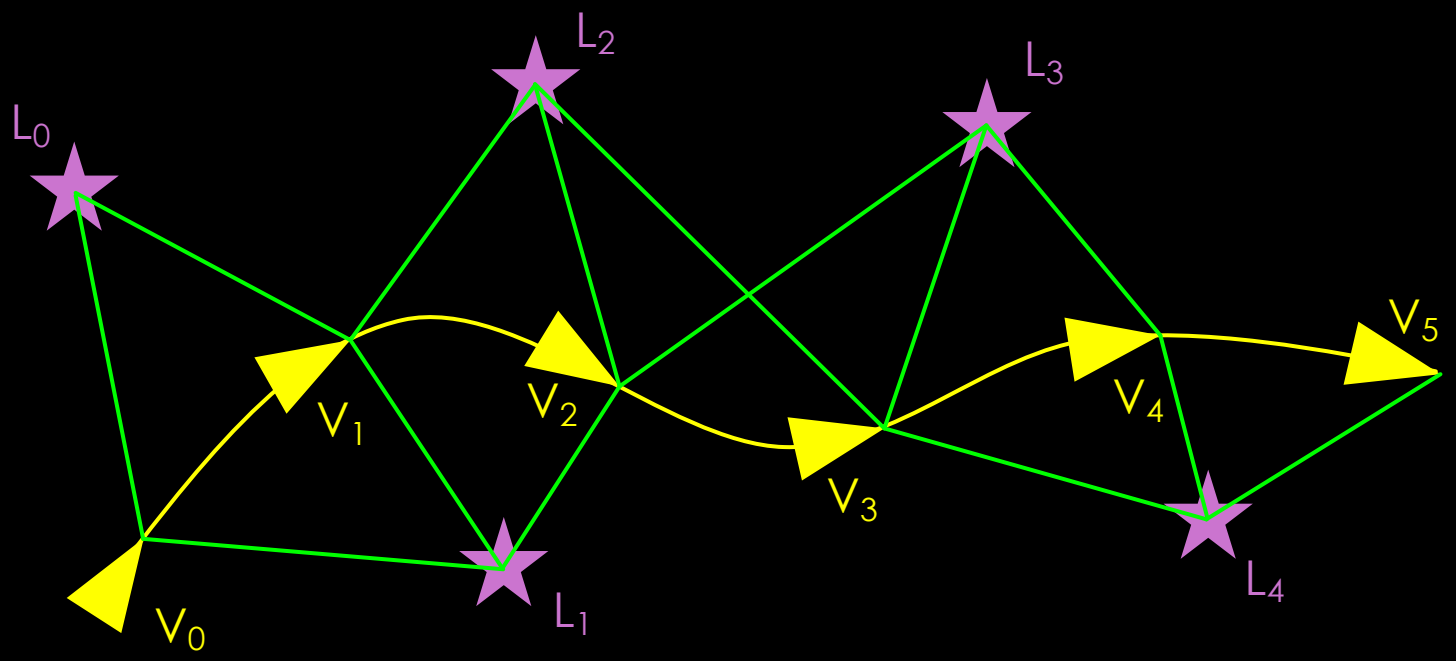
Models are built from:

- Point clouds (set of keypoints in space)
- + Appearance (what does each keypoint look like so we can recognize it)
- + Surface normal (what is the surface tangent around each point)
- Larger things: lines, planes, objects
- Meshes (usually triangulated, often with texture)
- Grids (2D or 3D): occupancy, signed distance function

Simultaneous Localisation And Mapping (SLAM)

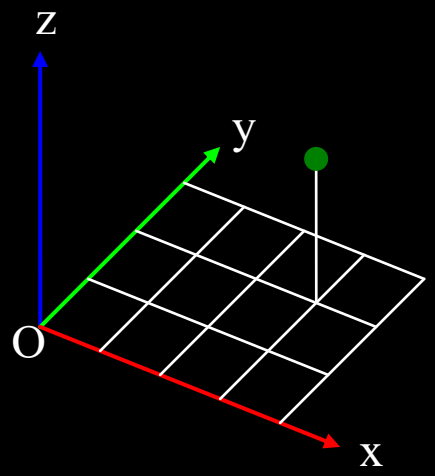


Simultaneous Localisation And Mapping (SLAM)



Point Cloud Models

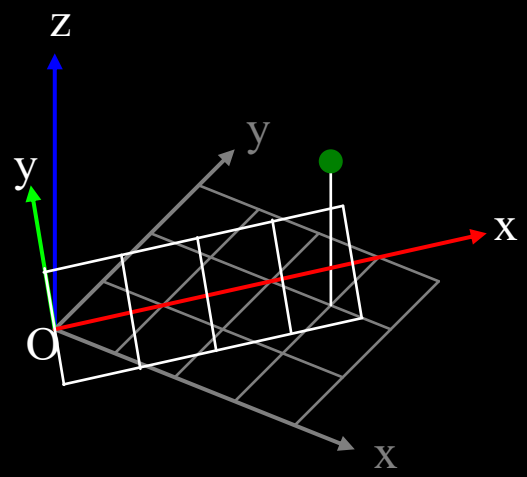
Position in space is represented using a vector for the x, y and z coordinates of a point:



For this to work, we have to agree where the origin is and the direction of each axis.

ROTATED AXES

If the axes point in different directions, then the point has different coordinates:



This rotation can be represented by a 3x3 matrix:

ROTATION MATRICES

ROTATIONS IN 3D ARE REPRESENTED BY 3X3 MATRICES

BUT NOT JUST ANY 3X3 MATRIX – WHAT ARE THE RULES?

$$R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}$$

LENGTHS MUST BE PRESERVED:

ANGLES MUST BE PRESERVED:

PRESERVING LENGTHS

PRESERVING ANGLES

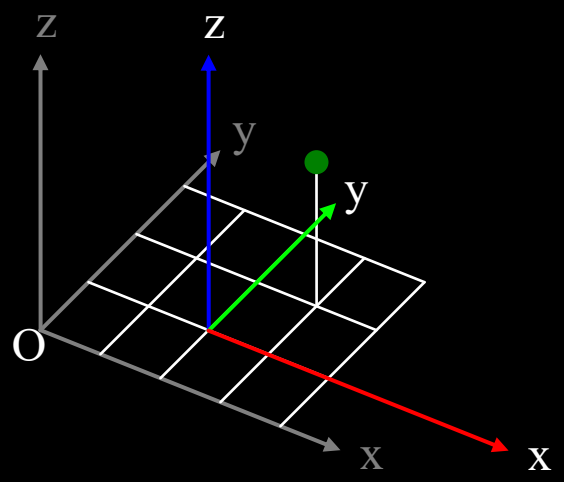
WHAT IS $R^T R$

THE MATRICES FOR ROTATIONS ABOUT X, Y AND Z AXES ARE VERY SIMPLE:

$$R_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) \\ 0 & \sin(\theta) & \cos(\theta) \end{bmatrix} \qquad R_y = \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{bmatrix}$$

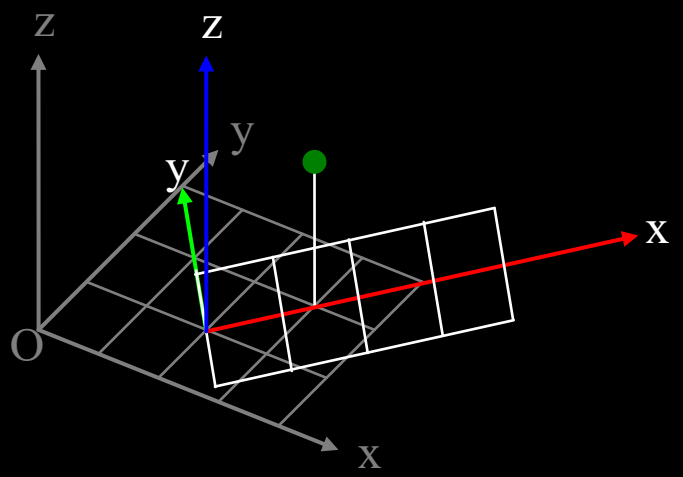
$$R_x = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

IF WE WANT TO PUT THE ORIGIN IN A DIFFERENT PLACE:



This shift (translation) can be represented by adding a vector:

WE CAN MOVE THE ORIGIN AND ROTATE THE FRAME AT THE SAME TIME



This can be represented using a rotation matrix and a vector:

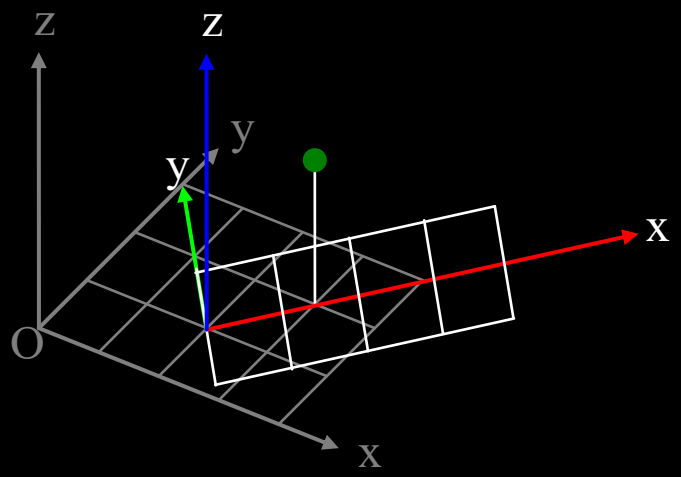
HOMOGENEOUS COORDINATES

POSITION VECTORS CAN BE EXTENDED TO 4 DIMENSIONS BY ADDING A 1 AT THE END. THIS REPRESENTATION OF POSITION IS CALLED *HOMOGENEOUS COORDINATES*.

THIS LETS US USE MATRIX MULTIPLICATION TO APPLY A TRANSLATION:

$$\begin{bmatrix} 1 & 0 & 0 & 2 \\ 0 & 1 & 0 & 3 \\ 0 & 0 & 1 & 4 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} =$$

HOMOGENEOUS COORDINATES



$$\begin{bmatrix} 0.707 & 0.707 & 0 & -2.121 \\ -0.707 & 0.707 & 0 & 0.707 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} 3 \\ 2 \\ 2 \\ 1 \end{pmatrix} =$$

$$M=\begin{bmatrix} \begin{bmatrix} & R & \end{bmatrix} & \begin{matrix} | \\ t \\ | \end{matrix} \\ \begin{matrix} 0 & 0 & 0 \end{matrix} & 1 \end{bmatrix} \qquad M^{-1} =$$

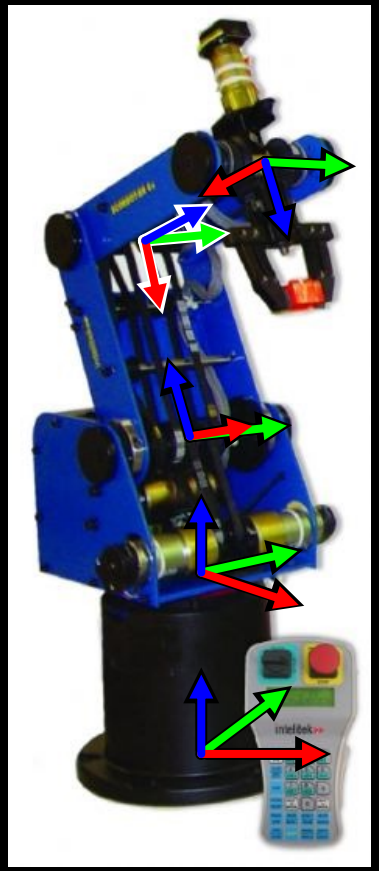
WHY DOES THIS MATTER?

THE ROTATION MATRIX AND THE TRANSLATION VECTOR DESCRIBE THE CHANGE IN COORDINATE FRAME.

THEY TELL US HOW TO CONVERT THE VECTOR THAT TURNS THE LOCATION OF A POINT IN ONE COORDINATE FRAME INTO THE VECTOR THAT GIVES THE LOCATION OF *THE SAME POINT* IN ANOTHER COORDINATE FRAME

IN ROBOTICS WE OFTEN HAVE *LOTS* OF COORDINATE FRAMES

ROBOTS HAVE LOTS OF COORDINATE FRAMES



wrist

elbow

shoulder

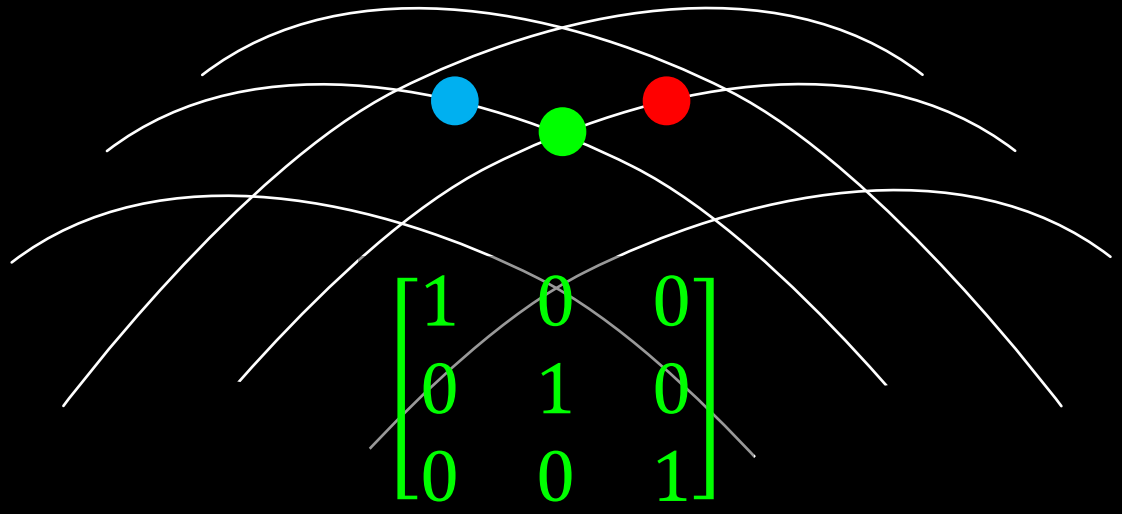
waist

world

ROTATION MATRICES MAKE A 3D “SURFACE” IN 9D SPACE

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0.98 & -0.17 \\ 0 & 0.17 & 0.98 \end{bmatrix}$$

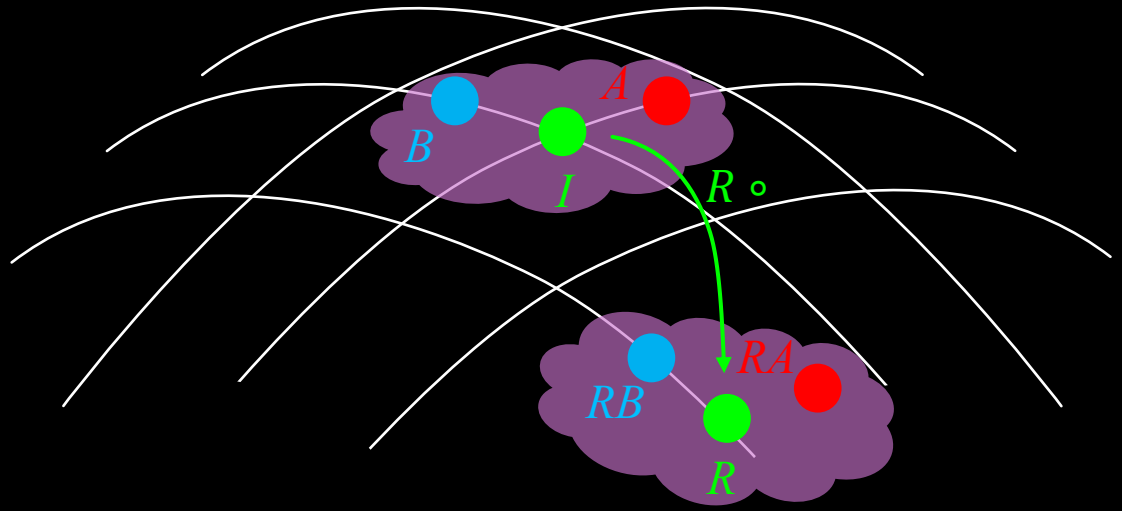
$$\begin{bmatrix} 0.98 & -0.17 & 0 \\ 0.17 & 0.98 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$



$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

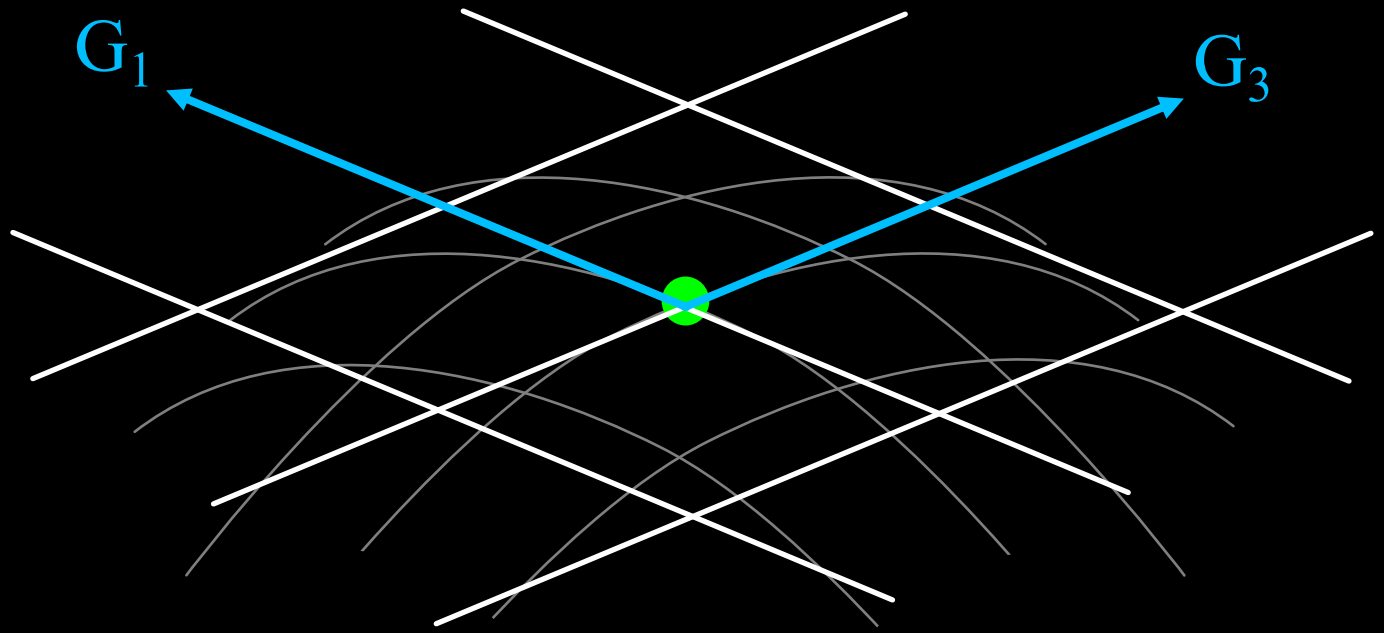
We can learn everything we need to know about the whole manifold by looking at the neighbourhood of the Identity matrix

WHY DO WE CARE SO MUCH ABOUT THE IDENTITY?



Because a neighbourhood of the Identity can be mapped to a neighbourhood of R by left multiplying by R

CAN FIT A TANGENT TO THE SURFACE AT THE IDENTITY



The tangent space of rotations is a 3D vector space
The basis axes are called generator matrices (G_1, G_2 , G_3)
Can add (infinitesimally) small amounts of these on to Identity and still have a rotation matrix

ROTATION MATRICES NEAR IDENTITY

Add small values to each element of the Identity matrix

$$R = \begin{bmatrix} 1+a & b & c \\ d & 1+e & f \\ g & h & 1+i \end{bmatrix} \qquad a, b, \ldots, i \quad \text{are small}$$

But $RR^T = I$

$$RR^T = \begin{bmatrix} 1+2a+a^2+b^2+c^2 & d+ad+b+be+cf & g+ga+bh+c+ci \\ d+ad+b+be+cf & d^2+1+2e+e^2+f^2 & dg+h+eh+f+fi \\ g+ga+bh+c+ci & dg+h+eh+f+fi & g^2+h^2+1+2i+i^2 \end{bmatrix} = I$$

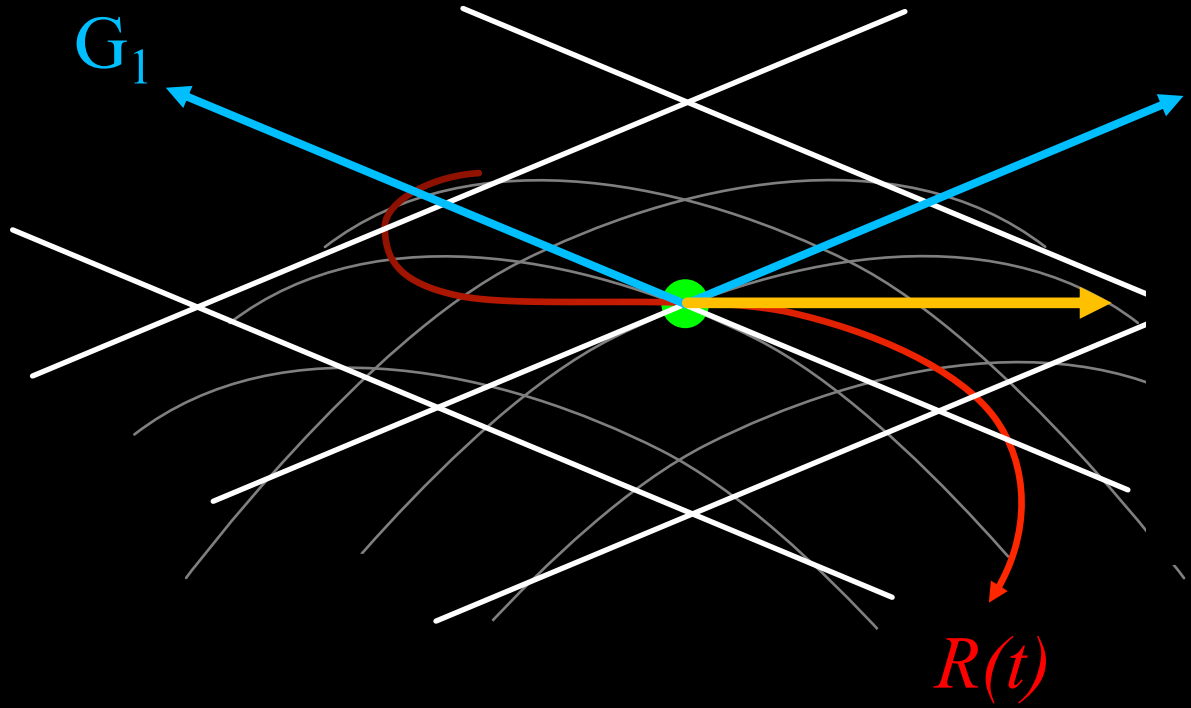
GENERATORS ARE A BASIS FOR THE TANGENT SPACE

$$G_1 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix}$$

$$G_2 = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \end{bmatrix}$$

$$G_3 = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

TANGENT SPACE IS ALSO THE SPACE OF DERIVATIVES

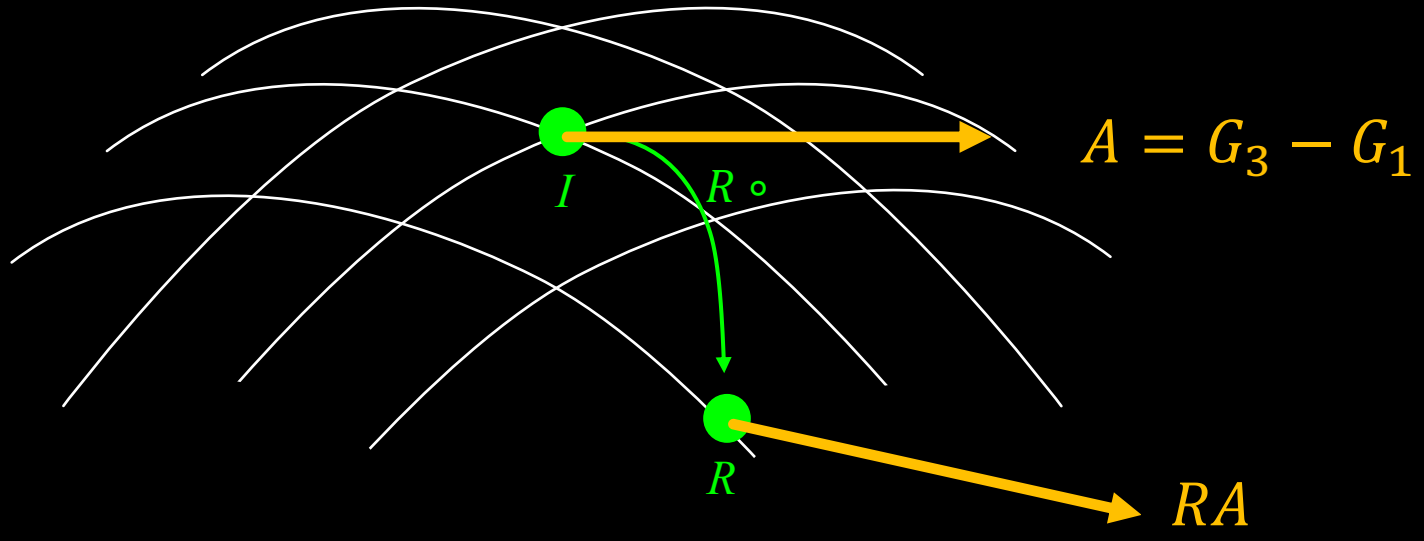


$$\frac{dR}{dt} = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 1 \\ 0 & -1 & 0 \end{bmatrix}$$

$$= G_3 - G_1$$

Derivatives are linear combinations of Generators
(any anti-symmetric matrix)

GROUP ELEMENTS CAN MOVE DERIVATIVES TOO

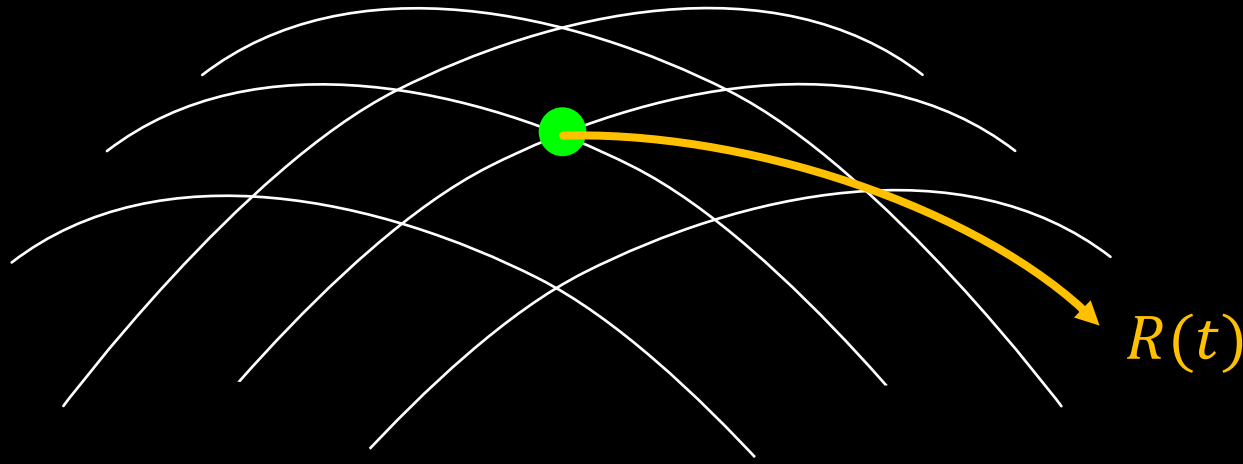


WALKING IN A STRAIGHT LINE (GEODESIC)

Start at the Identity ($R(0) = I$)

Choose a direction (derivative = A)

Move the derivative with us as we walk (derivative = RA)



$$\frac{dR}{dt} = RA$$

SIMPLE EXAMPLE:

$$e^{\begin{bmatrix} 0 & -\theta \\ \theta & 0 \end{bmatrix}} =$$

WHAT IS A GROUP, LIE GROUP, LIE ALGEBRA?

EXPONENTIALS OF MATRICES

This can be interpreted using a different expansion for exp

$$e^{A} = \lim_{n \to \infty} \left(I + \frac{A}{n} \right)^{n}$$

As n becomes large A/n becomes infinitesimal

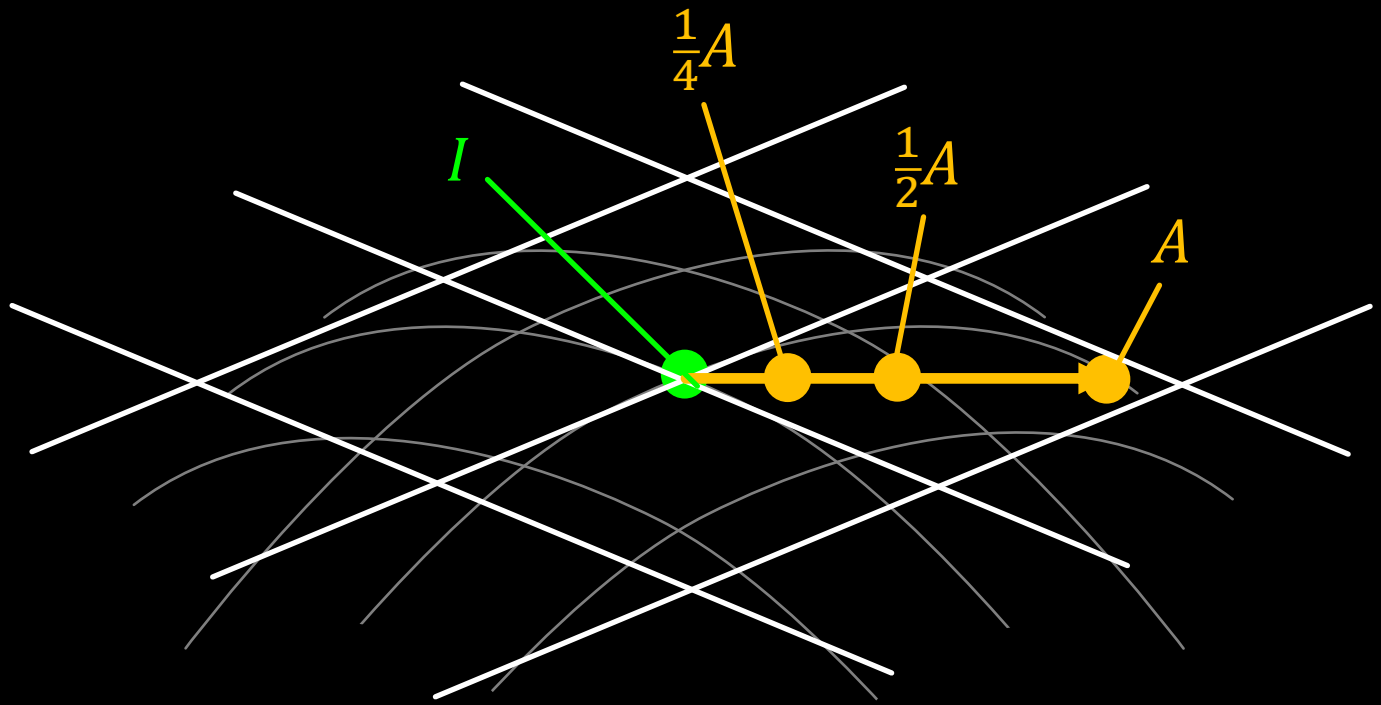
So I+A/n becomes closer to a rotation matrix

(nearer to Identity in tangent plane)

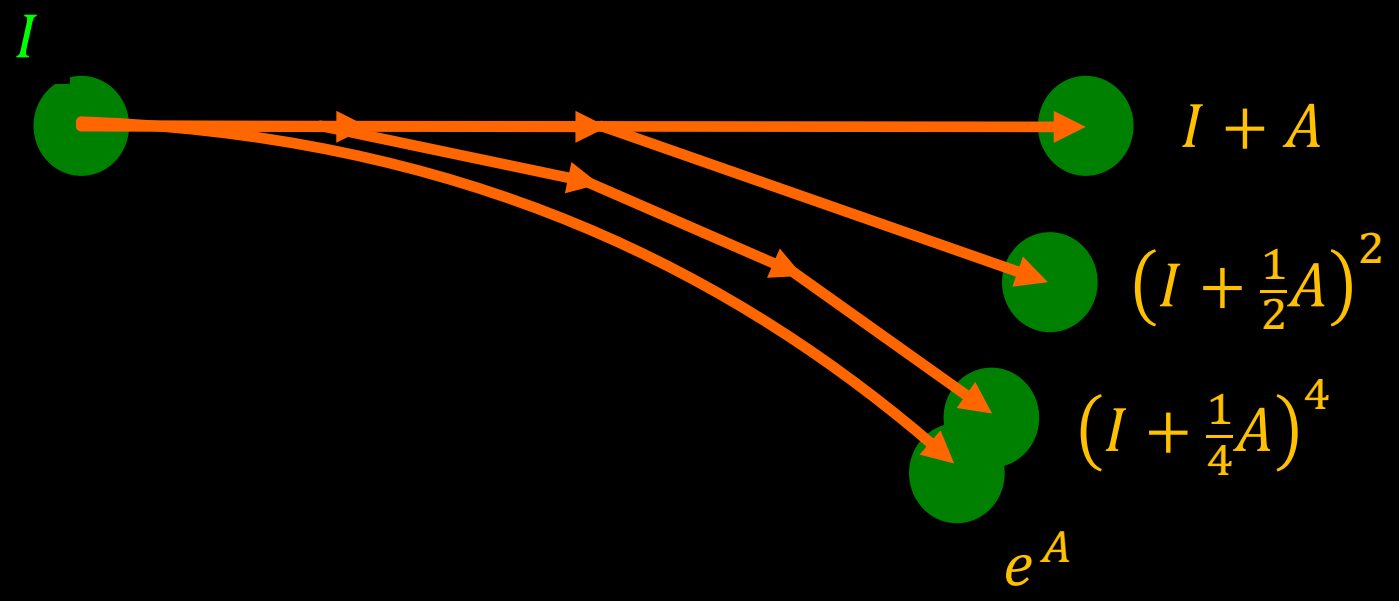
Raising to power n is just multiplying lots of rotations together

EXPONENTIALS OF MATRICES

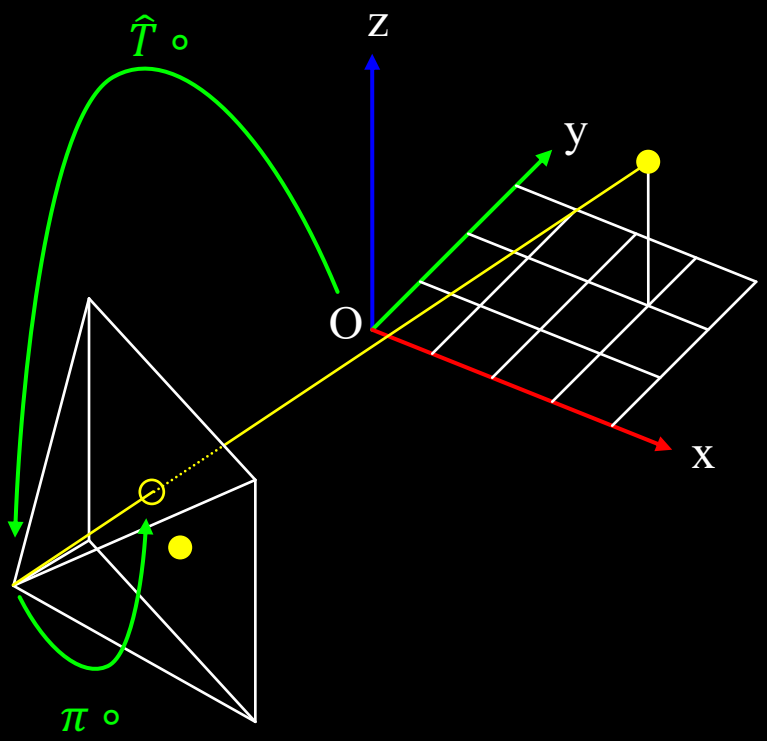
As smaller fractions of M are taken, we get closer to the manifold of rotations



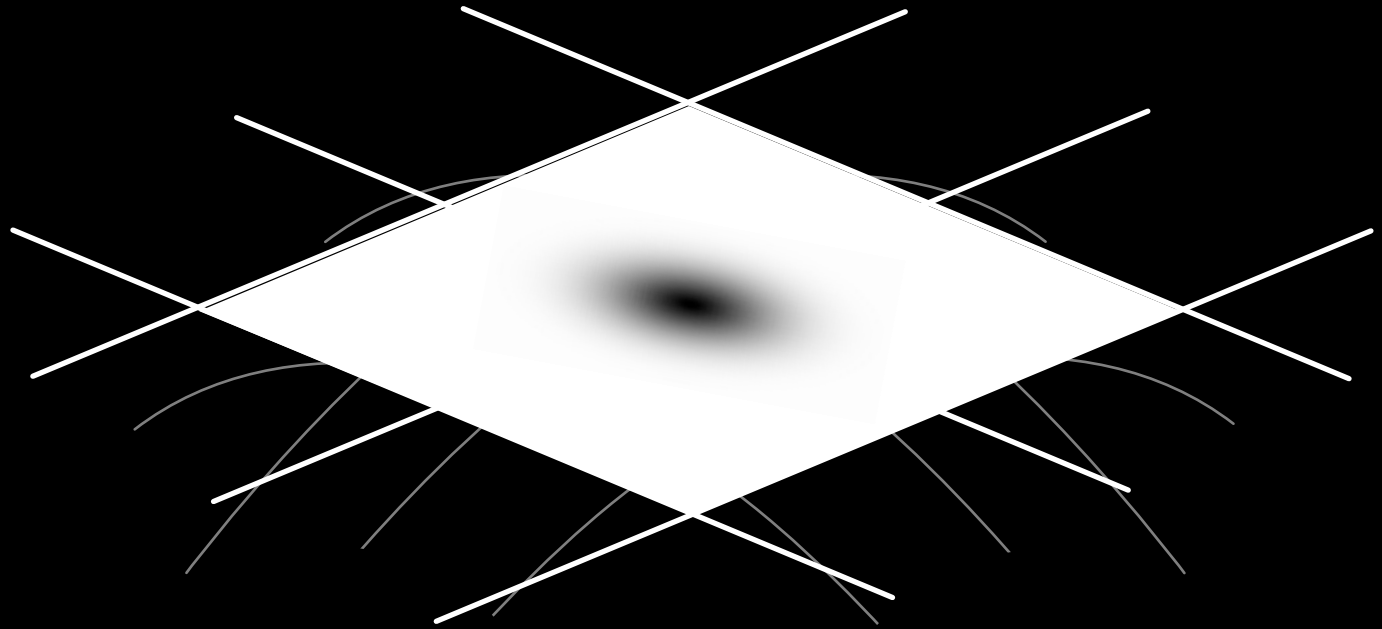
EXPONENTIALS OF MATRICES



HOW DO WE USE THIS?

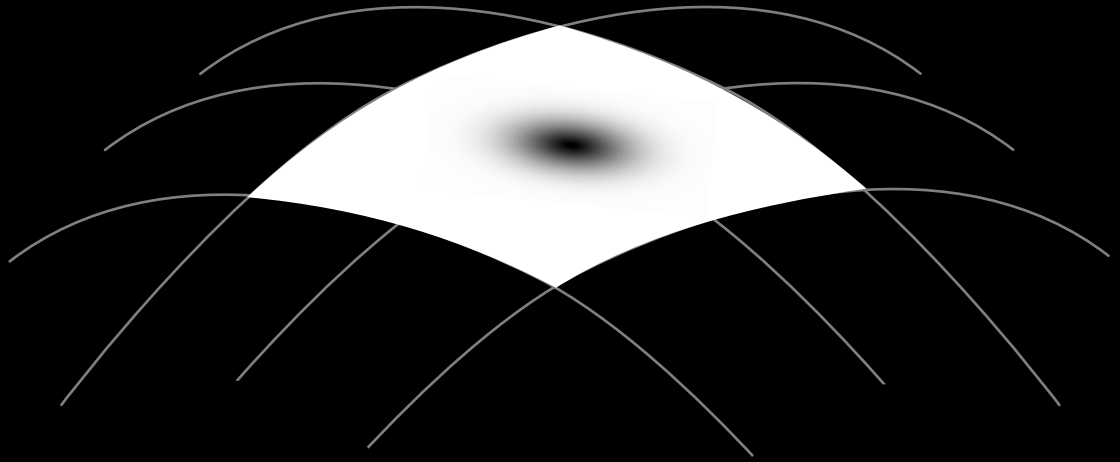


WHAT ABOUT UNCERTAINTIES?
CAN PLACE A GAUSSIAN PDF ON THE TANGENT SPACE…

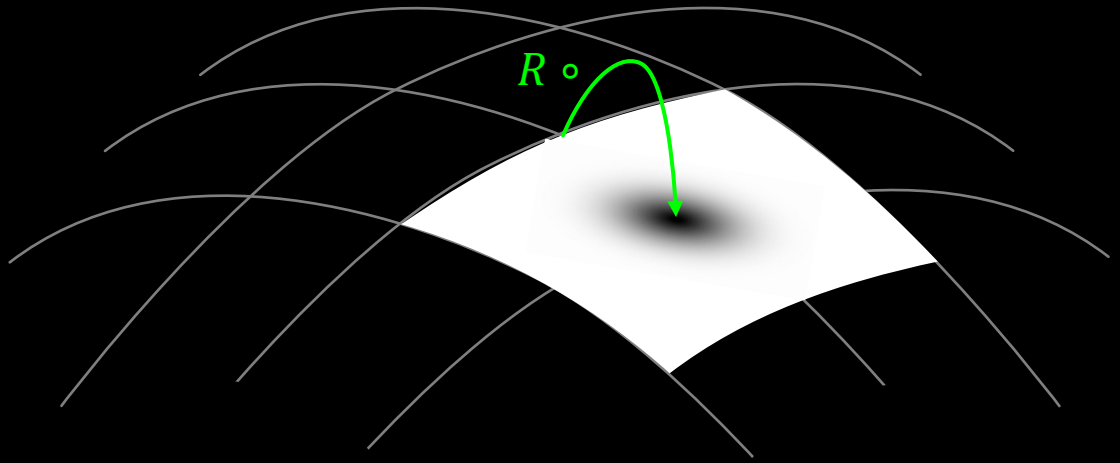


EXPONENTIAL MAP PROJECTS ONTO THE GROUP

Draw samples from the tangent space, apply exponential map to each sample



AND MOVE THE CENTRE OF THE PDF BY LEFT (OR RIGHT) MULTIPLYING BY SOME GROUP ELEMENT

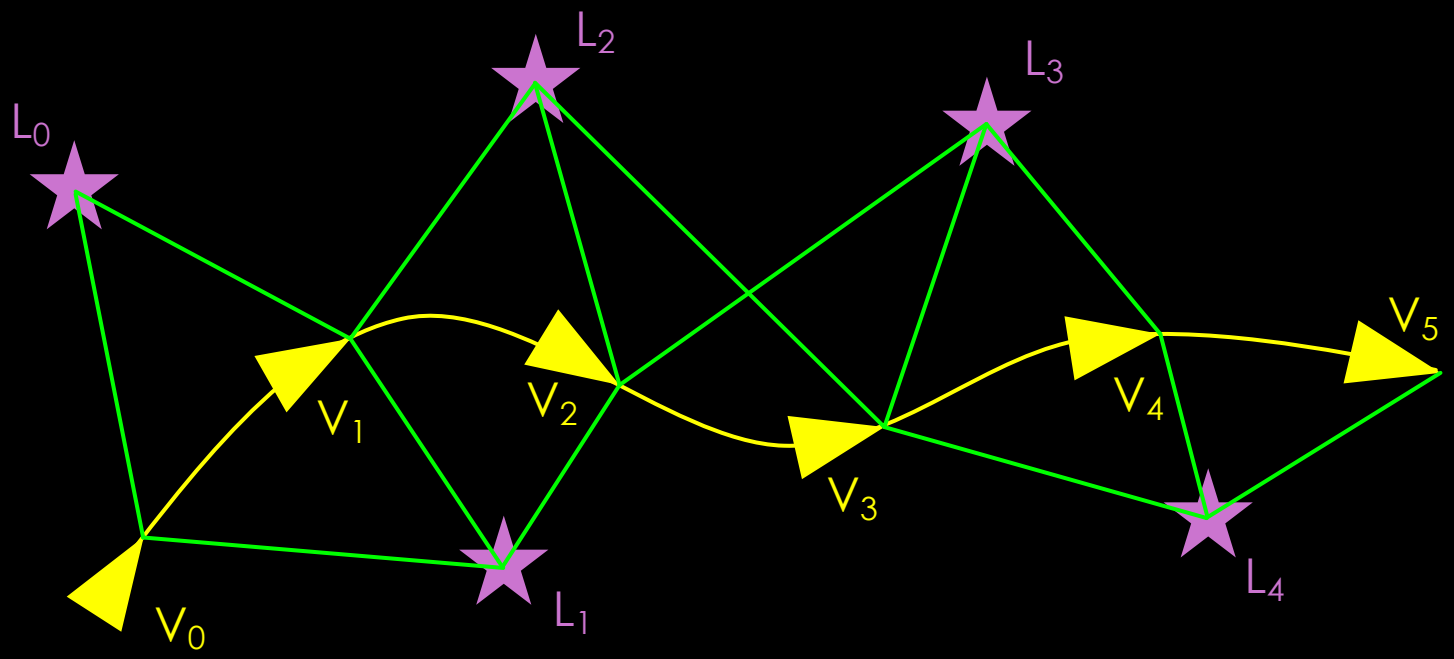


RVSS 2026

Part II: Classical SLAM

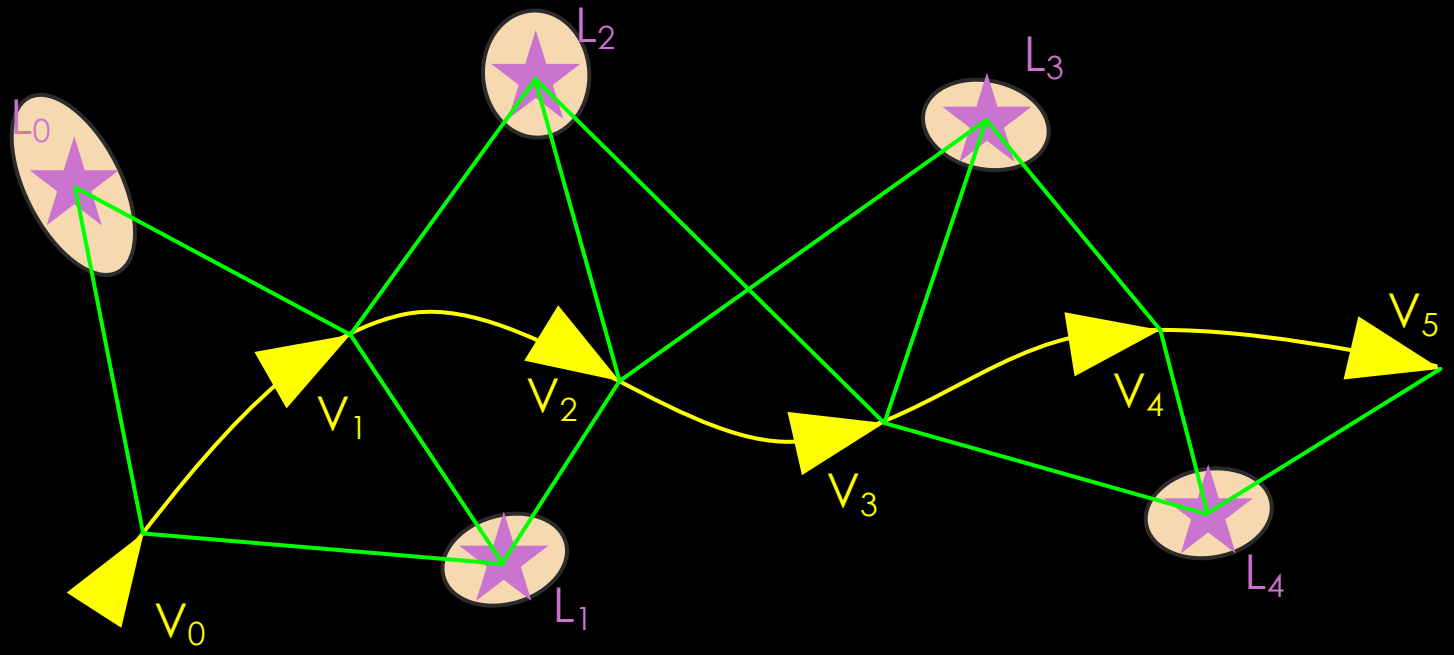
Prof Tom Drummond

Simultaneous Localisation And Mapping (SLAM)



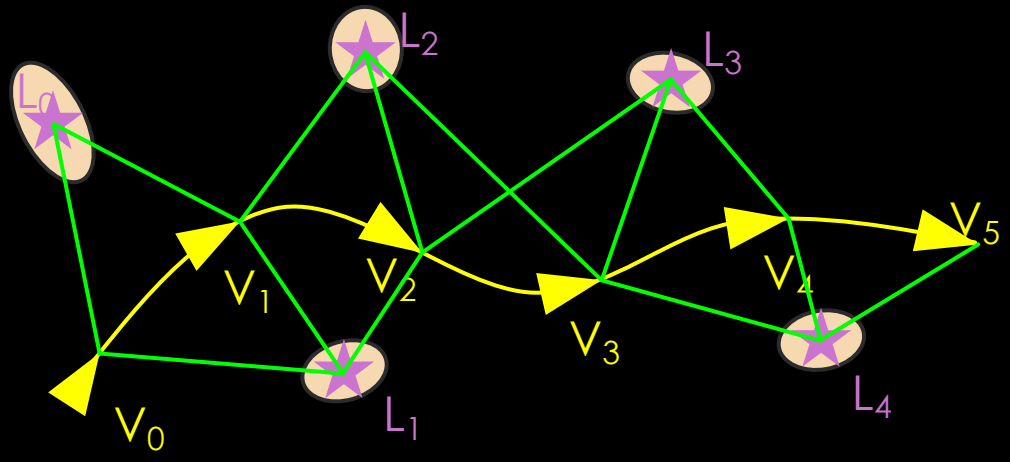
How Many Degrees of Freedom in this figure?
(Assuming we are operating in 3 dimensions)

Simultaneous Localisation And Mapping (SLAM)



Represent uncertainty using
multivariate Gaussian density functions

EKF SLAM

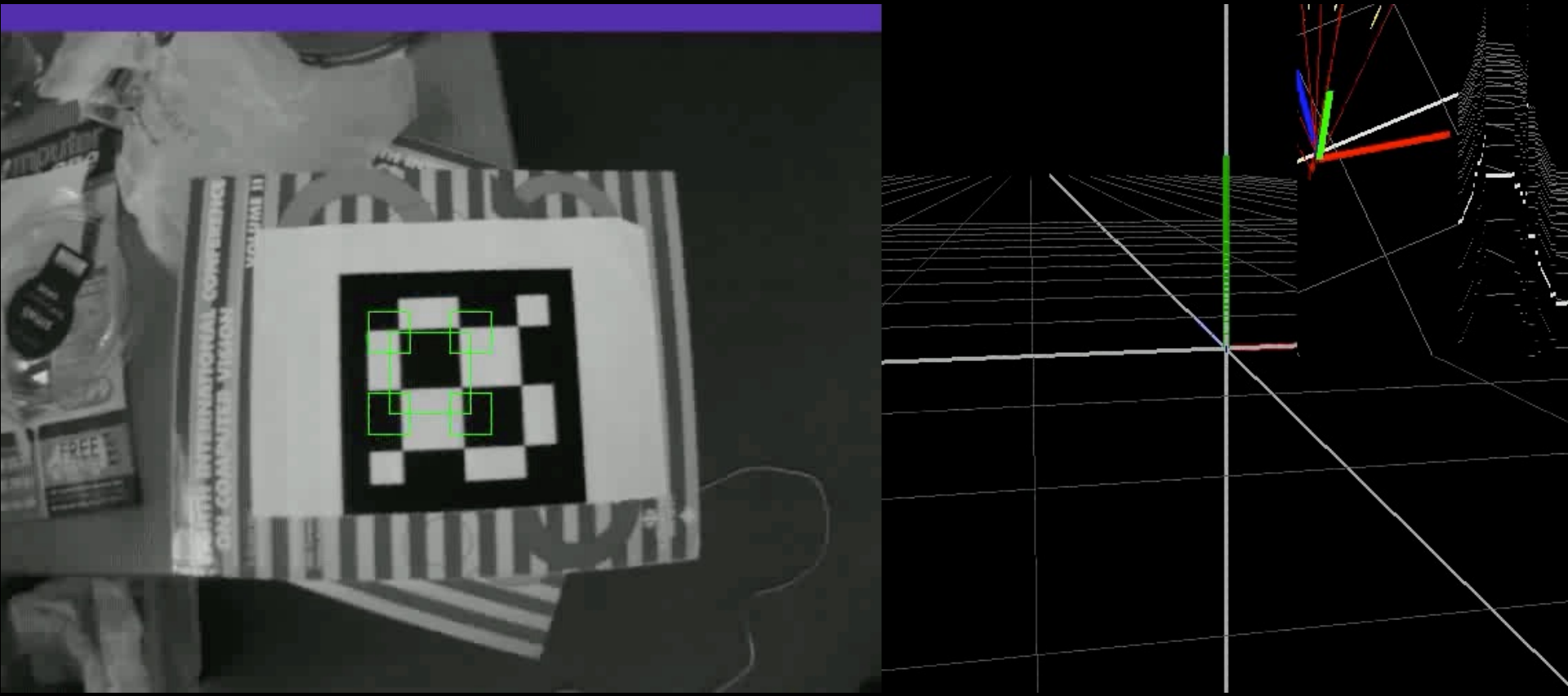


State Variables:

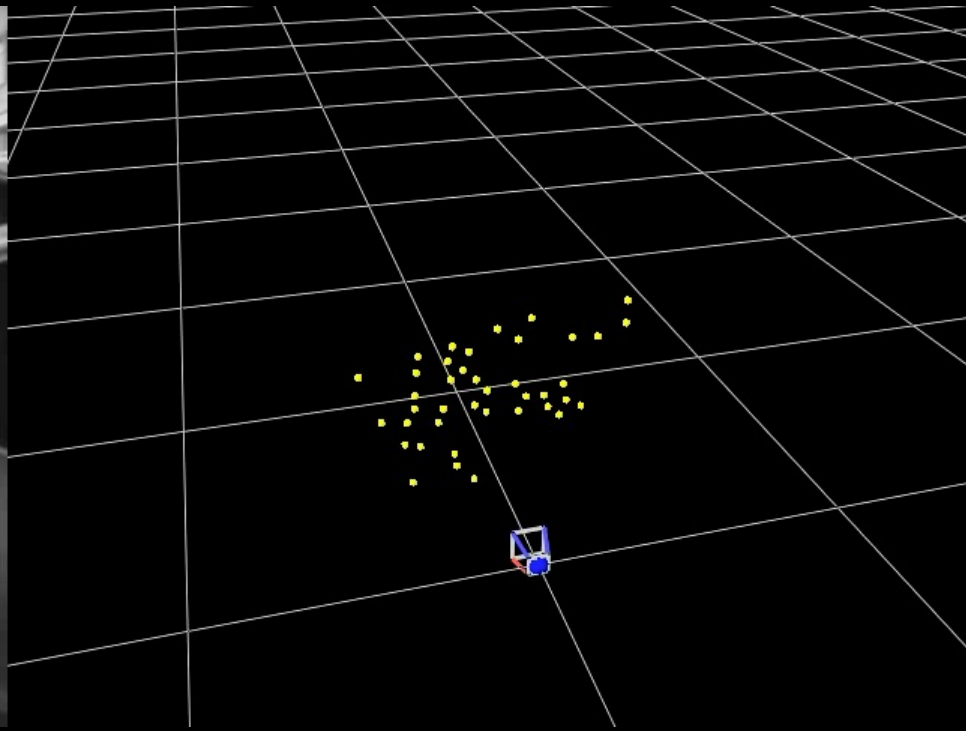
- Current vehicle pose
- Landmark positions

Step 1: Predict (calculate where I should see the landmarks from my estimate)
Step 2: Measure (observe landmarks and calculate error (aka innovation))
Step 3: Update (Calculate pdf of the posterior pose and landmarks)
Step 4: Time evolution (estimate vehicle position from odometry / velocity + add noise)

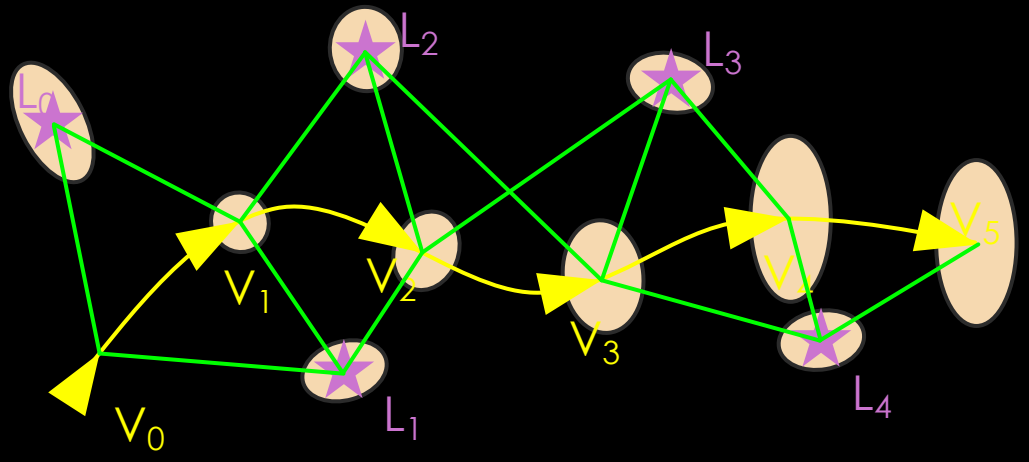
Simultaneous Localisation And Mapping (SLAM)



Simultaneous Localisation And Mapping (SLAM)



GRAPH SLAM



State Variables:

- All vehicle poses
- Landmark positions

Step 1: Measure (observe landmarks from new location)
Step 2: Add to long list of all measurements from all vehicle poses
Step 3: Predict all measurements from all vehicle poses
Step 4: Compute errors between measurements and predictions
Step 5: Compute Jacobian of these errors
Step 6: Use Gauss-Newton to update the estimates of all vehicle poses and all landmarks

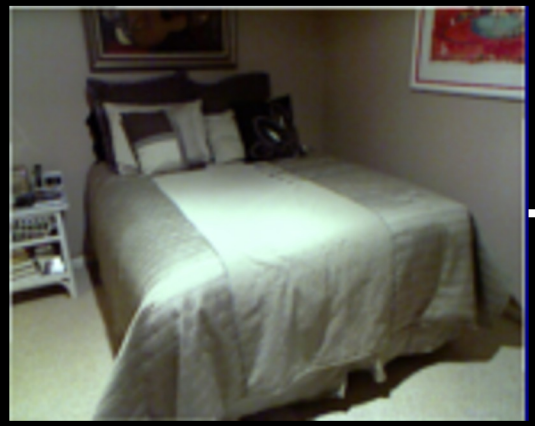
Gauss Newton Update

RVSS 2026

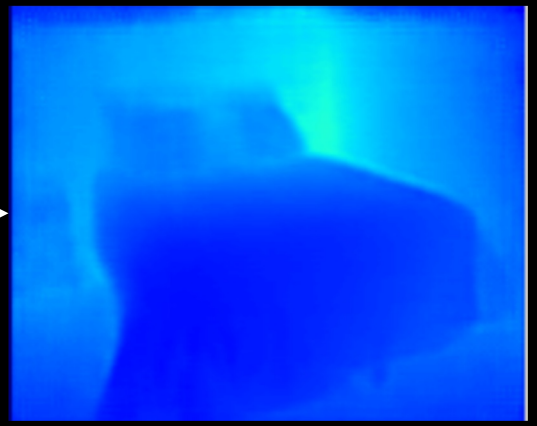
Geometry Part III: Deep Learning for SLAM

Prof Tom Drummond

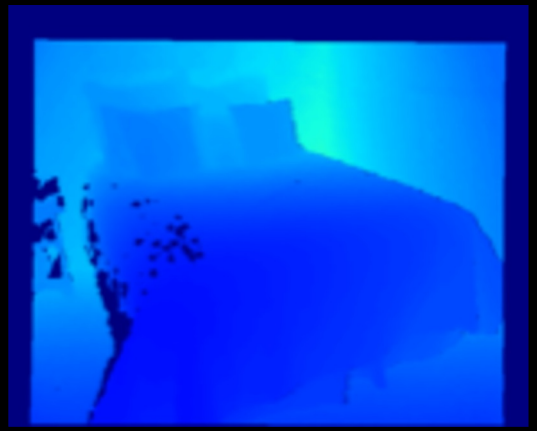
Can use deep learning to estimate depth from a single image



Depth estimation network

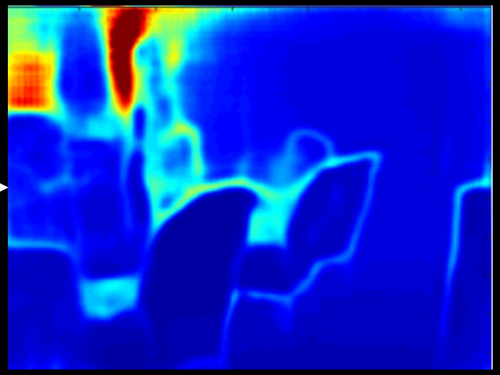


Can minimize L_2 loss to ground truth:

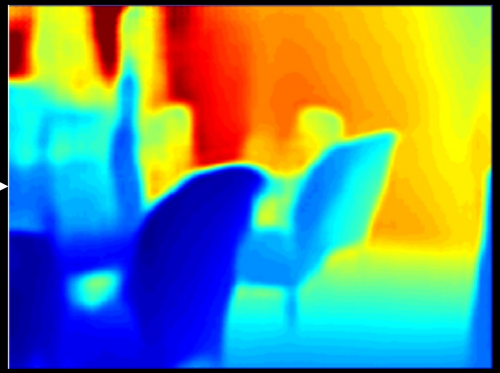


Better to learn uncertainty as well as depth estimates

Depth estimation network



Depth (μ)

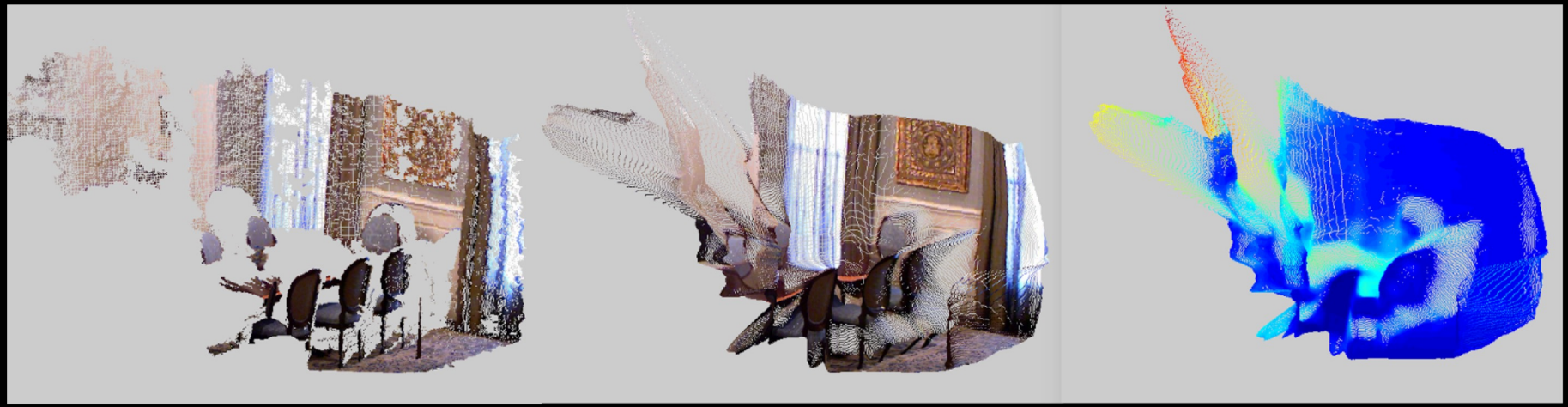


Uncertainty (σ)

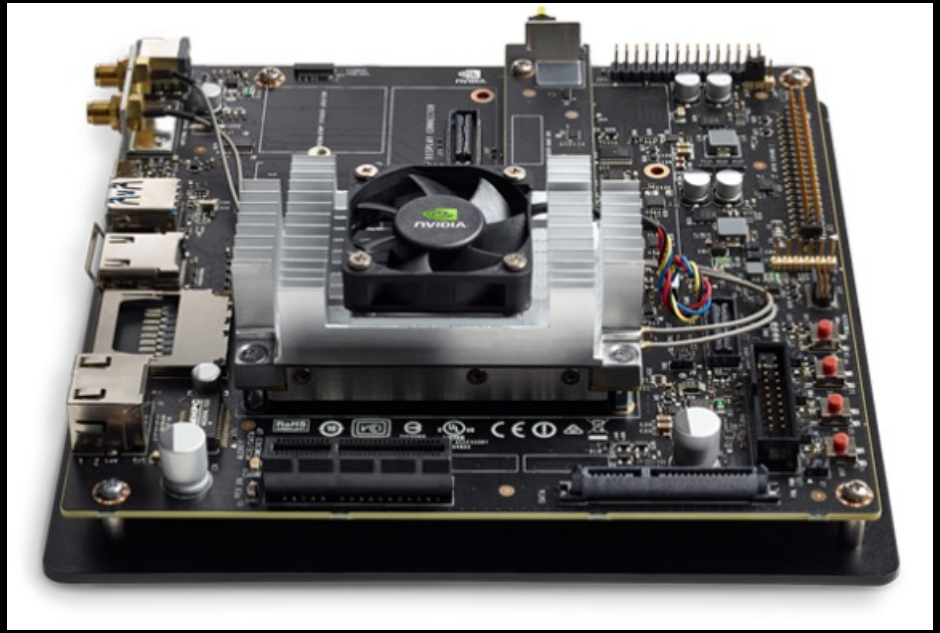
Mean (μ) and sigma (σ) imply a pdf

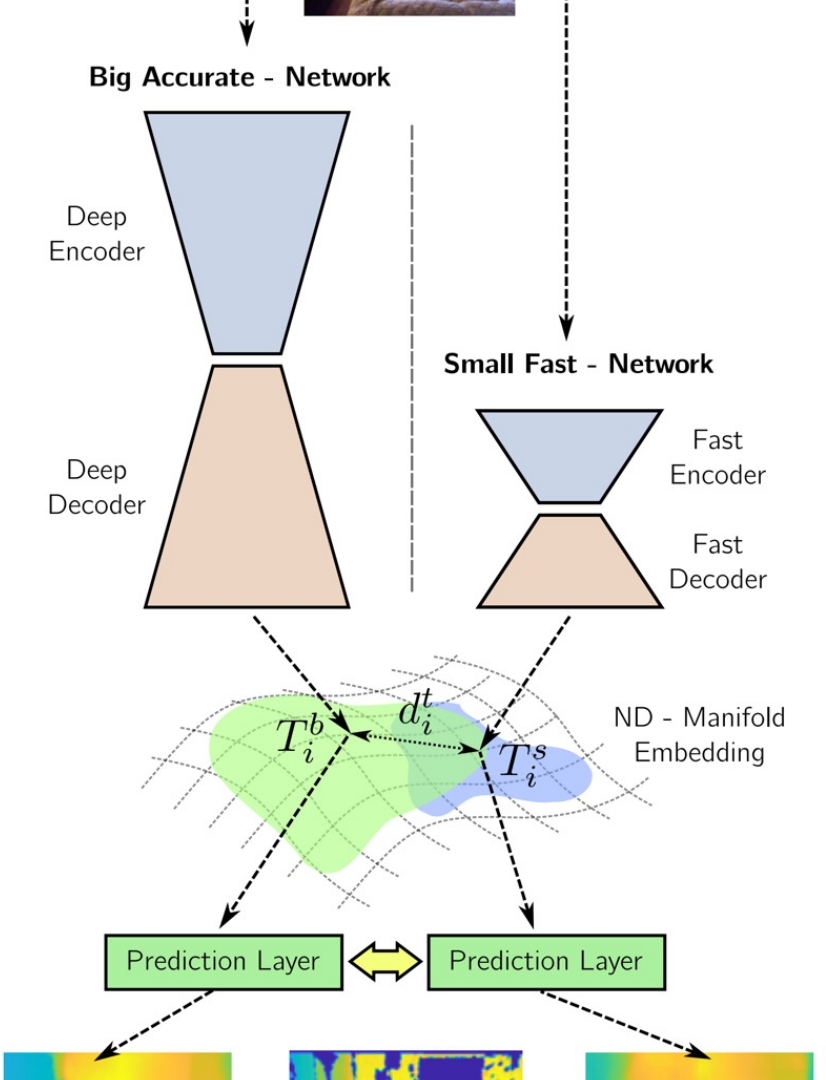
Use this to minimize log loss to ground truth

Uncertainty maps reflect the errors



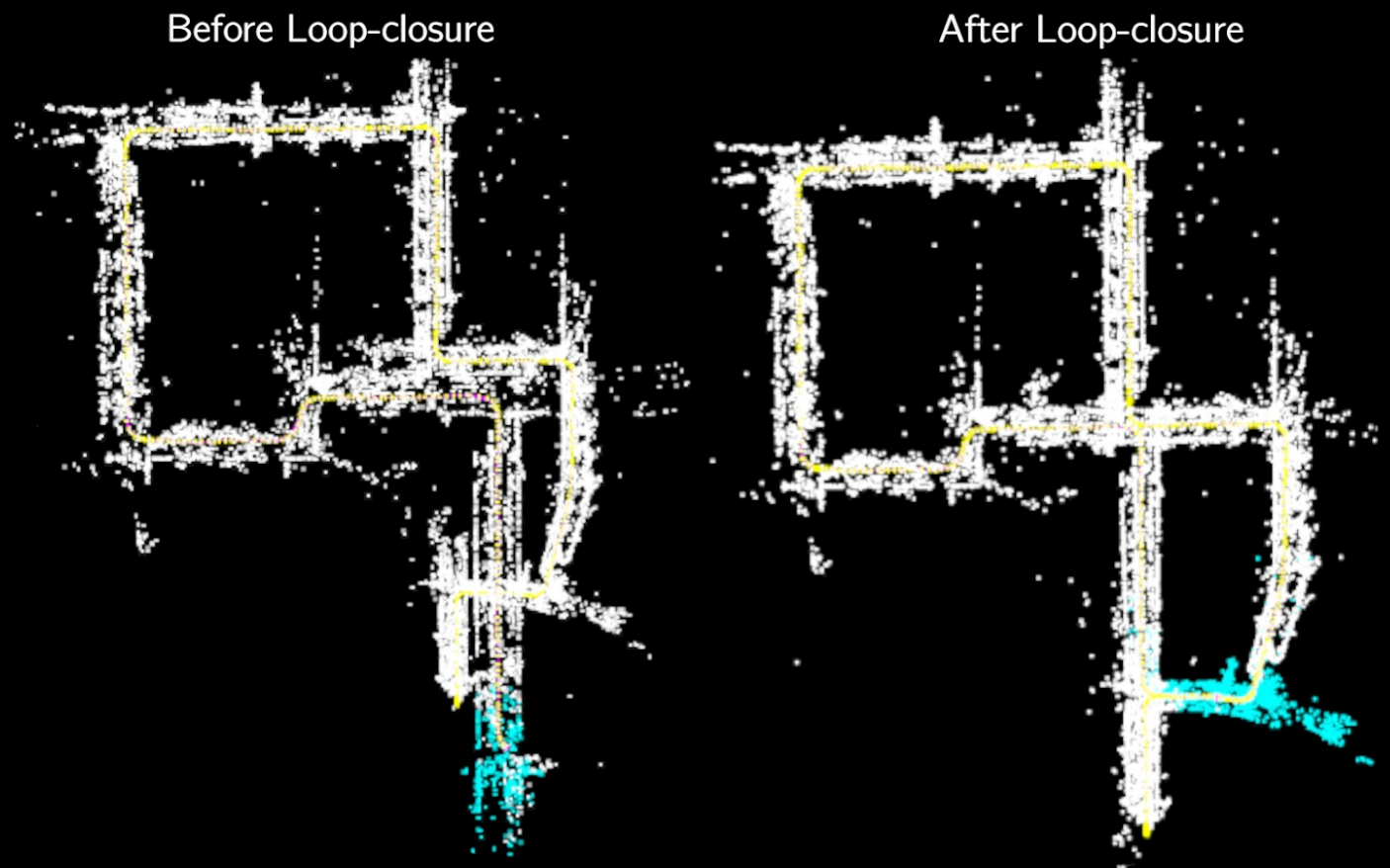
Speeding it up for robotics





Can use depth to improve ORB SLAM

Monocular Only

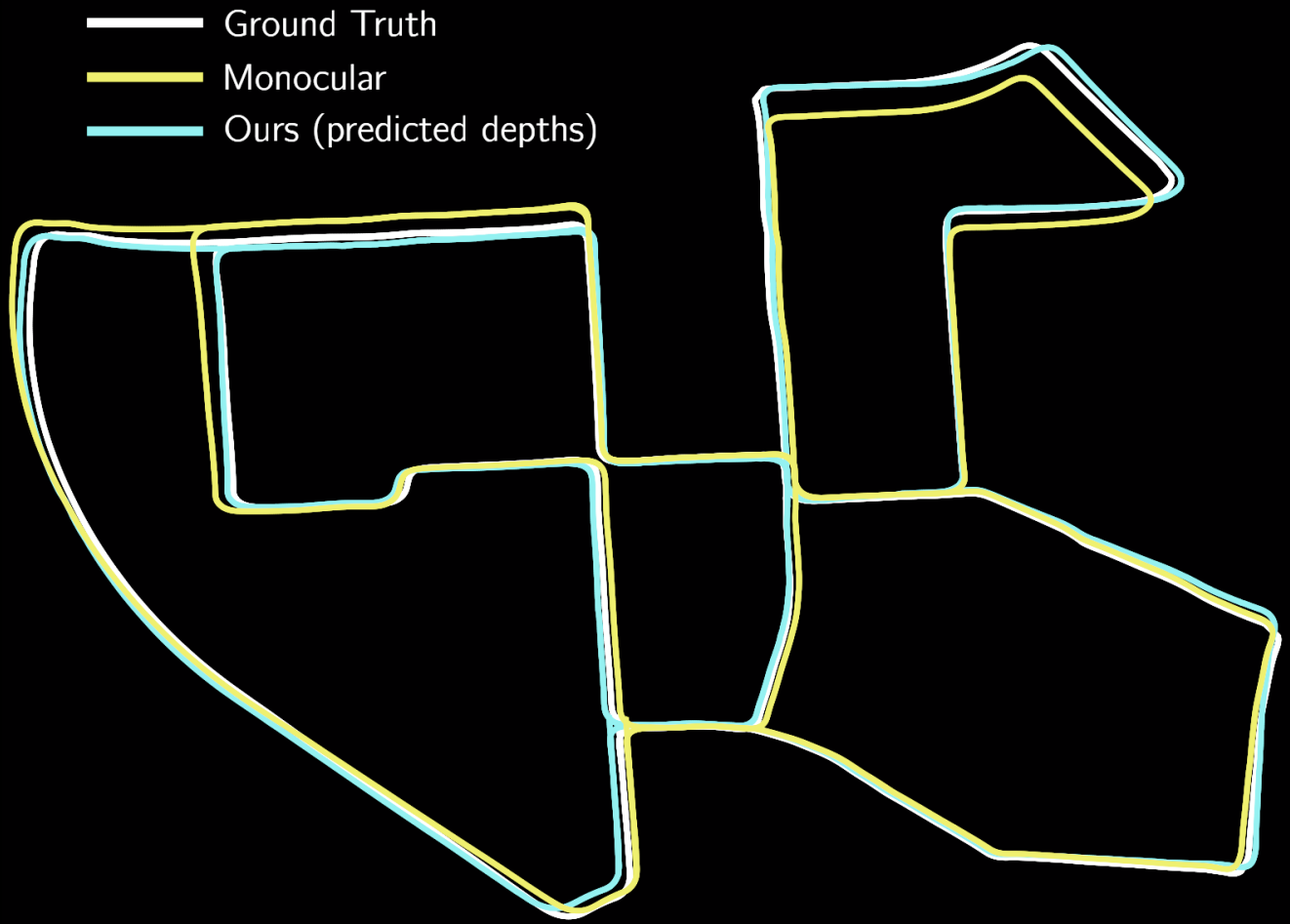


Using Depth Predictions

Before Loop-closure

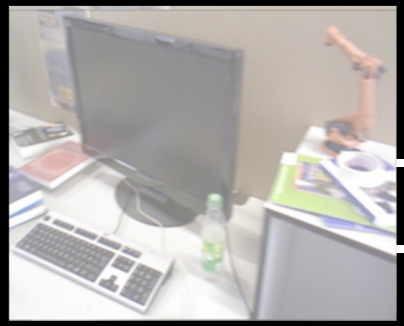
After Loop-closure

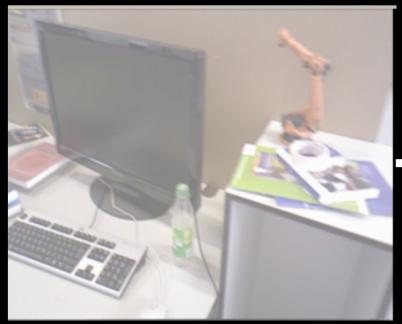
Can use depth to improve ORB SLAM



But we want more deep learning!

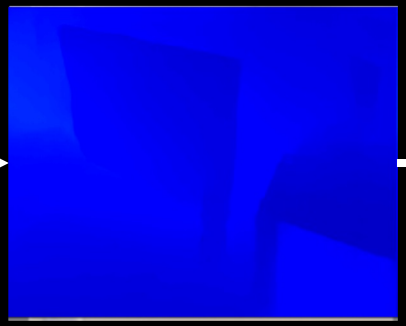
Can also compute flow from two images and use this to compute camera motion

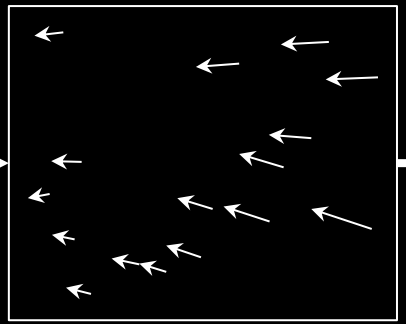




Depth estimation network

Flow estimation network

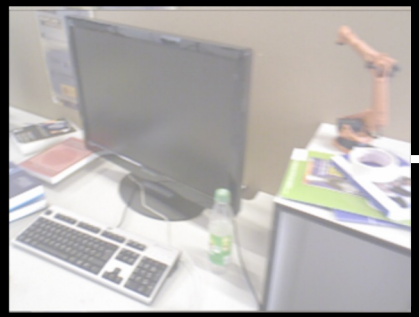


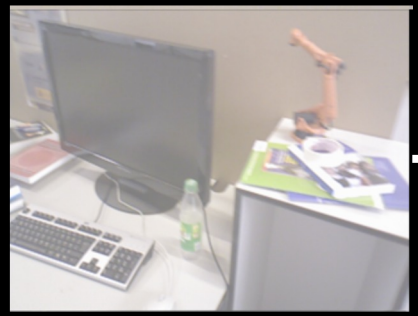


Motion estimation via geometry

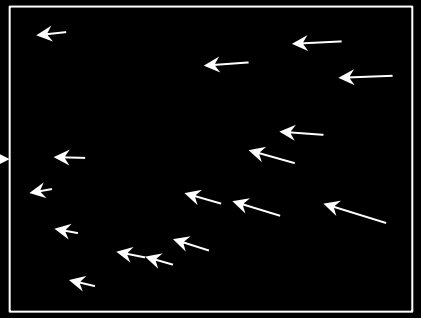
Motion and rotation

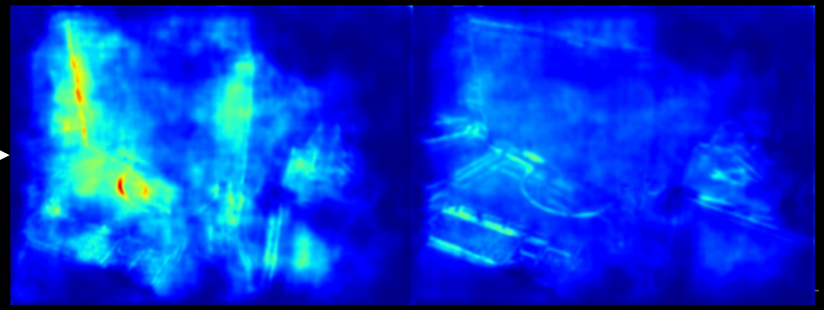
Better if uncertainty in flow is incorporated too





Flow estimation network



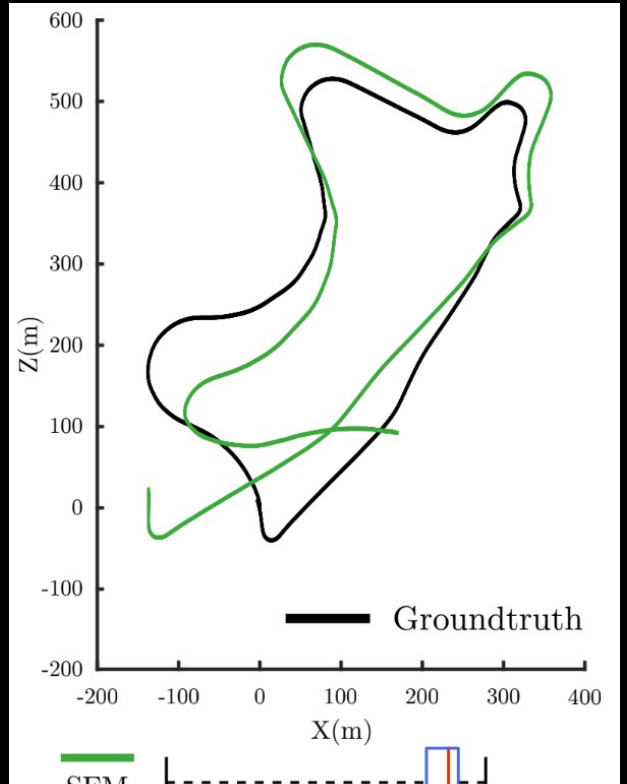


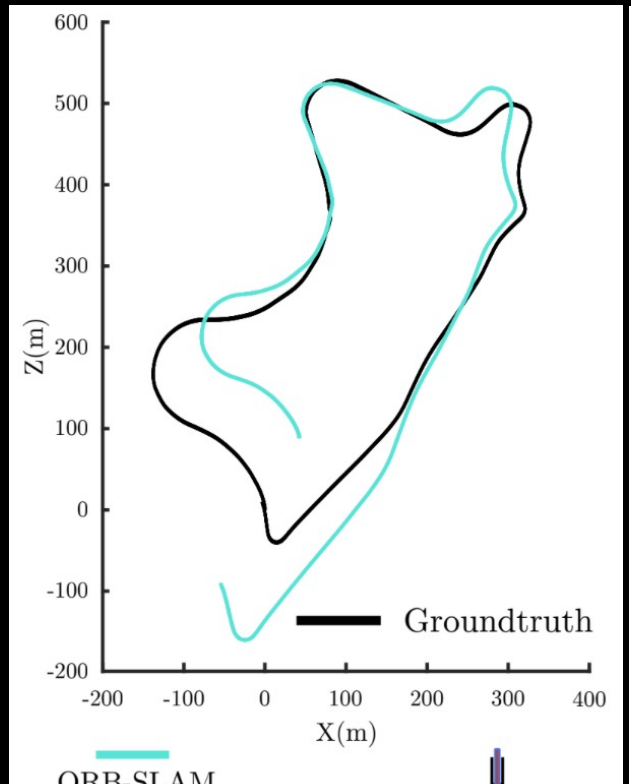
X

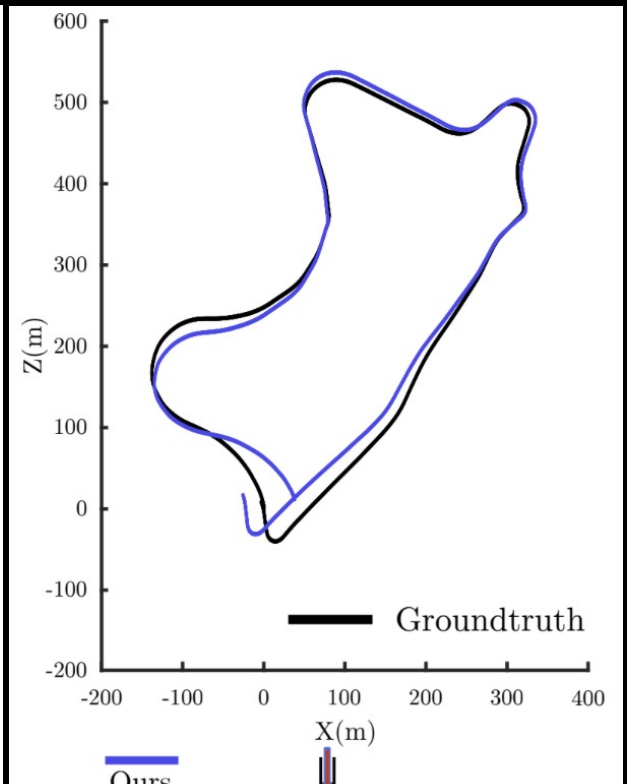
Y

Confidence in flow estimate

VISUAL ODOMETRY (JUST PRIOR TO LOOP CLOSURE)







Interestingly, system learns to recognise independently moving objects and give them less confidence

DUST3R

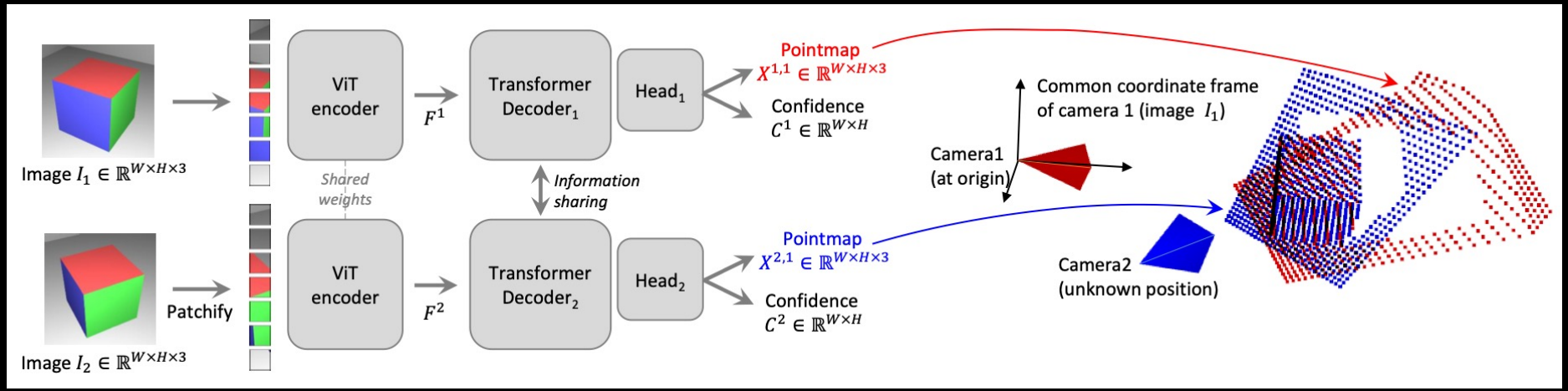
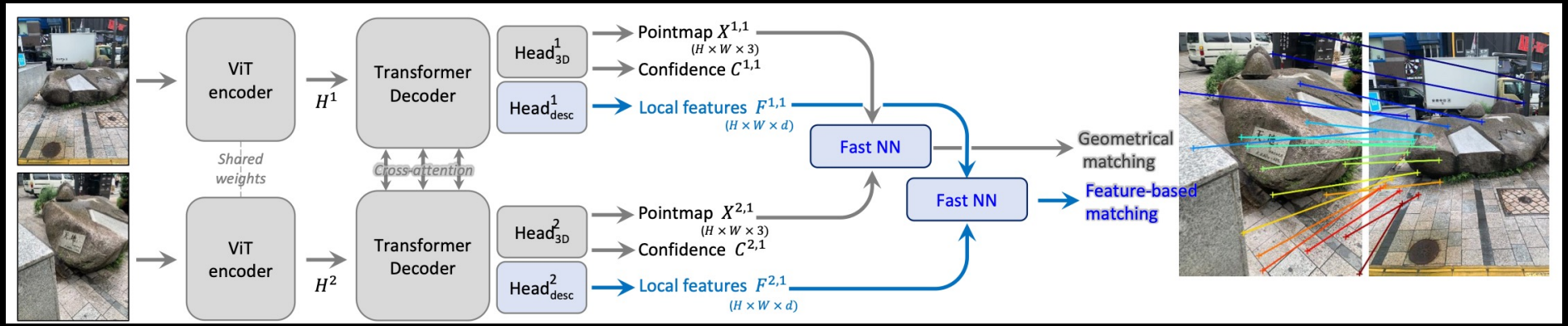
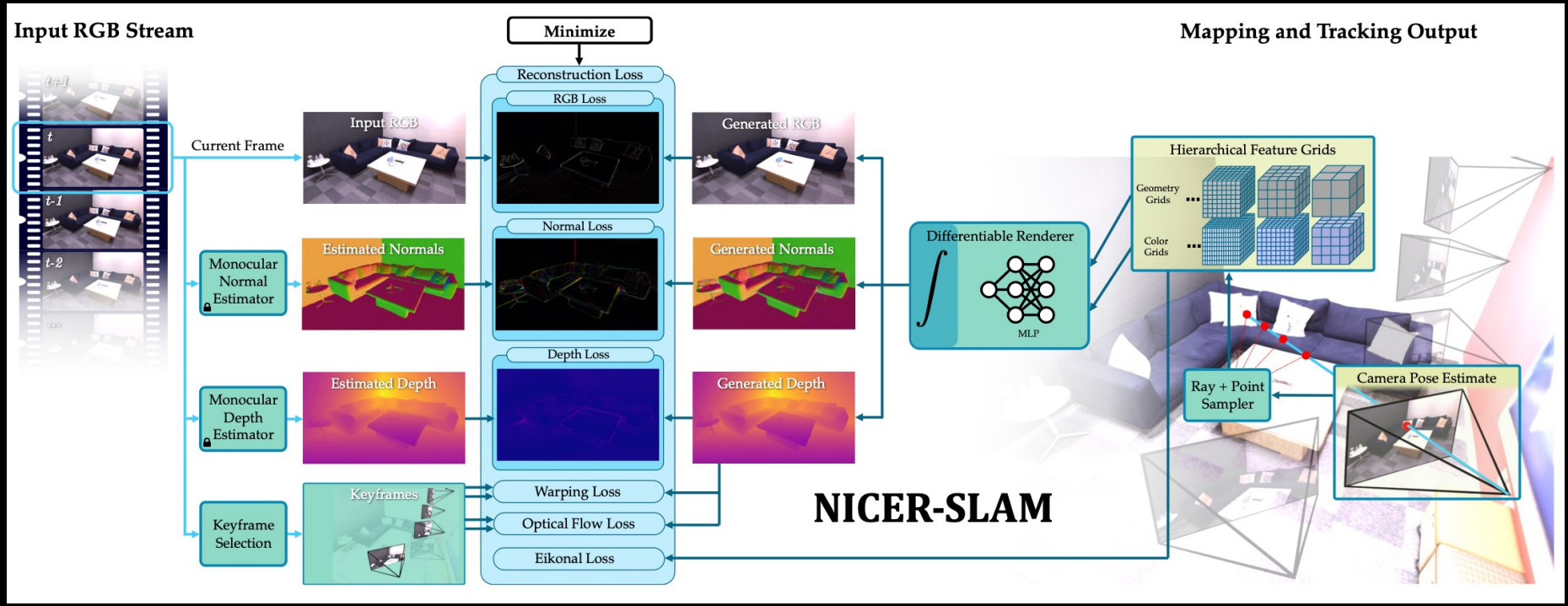


Image $I_1 \in \mathbb{R}^{W \times H \times 3}$
ViT encoder
F^1
Transformer Decoder$_1$
Head$_1$
Pointmap
$X^{1,1} \in \mathbb{R}^{W \times H \times 3}$
Confidence
$C^1 \in \mathbb{R}^{W \times H}$
Shared weights
Information sharing
Image $I_2 \in \mathbb{R}^{W \times H \times 3}$
Patchify
ViT encoder
F^2
Transformer Decoder$_2$
Head$_2$
Pointmap
$X^{2,1} \in \mathbb{R}^{W \times H \times 3}$
Confidence
$C^2 \in \mathbb{R}^{W \times H}$
Common coordinate frame of camera 1 (image I_1)
Camera1 (at origin)
Camera2 (unknown position)

MAST3R


ViT encoder
H^1
Transformer Decoder
Shared weights
Cross-attention
ViT encoder
H^2
Transformer Decoder
Head^1_{3D}
Head^1_{desc}
Head^2_{3D}
Head^2_{desc}
Pointmap $X^{1,1}$ $(H \times W \times 3)$
Confidence $C^{1,1}$
Local features $F^{1,1}$ $(H \times W \times d)$
Pointmap $X^{2,1}$ $(H \times W \times 3)$
Confidence $C^{2,1}$
Local features $F^{2,1}$ $(H \times W \times d)$
Fast NN
Fast NN
Geometrical matching
Feature-based matching

NICER-SLAM


Input RGB Stream
Minimize
Mapping and Tracking Output
t+1
t
t-1
t-2
Current Frame
Input RGB
Reconstruction Loss
RGB Loss
Generated RGB
Monocular Normal Estimator
Estimated Normals
Normal Loss
Generated Normals
Monocular Depth Estimator
Estimated Depth
Depth Loss
Generated Depth
Keyframe Selection
Keyframes
Warping Loss
Optical Flow Loss
Eikonal Loss
Differentiable Renderer
MLP
Hierarchical Feature Grids
Geometry Grids
Color Grids
Ray + Point Sampler
Camera Pose Estimate
NICER-SLAM