Geometric Primitives & Transformations

Juan Carlos Niebles and Adrien Gaidon

Reference: Szeliski 2.1
# What is the most popular topic at CVPR?

<table>
<thead>
<tr>
<th>Publication</th>
<th>h5-index</th>
<th>h5-median</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Nature</td>
<td>467</td>
<td>707</td>
</tr>
<tr>
<td>2. The New England Journal of Medicine</td>
<td>439</td>
<td>876</td>
</tr>
<tr>
<td>3. Science</td>
<td>424</td>
<td>665</td>
</tr>
<tr>
<td><strong>4. IEEE/CVF Conference on Computer Vision and Pattern Recognition</strong></td>
<td><strong>422</strong></td>
<td><strong>681</strong></td>
</tr>
<tr>
<td>5. The Lancet</td>
<td>368</td>
<td>688</td>
</tr>
<tr>
<td>6. Nature Communications</td>
<td>349</td>
<td>456</td>
</tr>
<tr>
<td>7. Advanced Materials</td>
<td>326</td>
<td>415</td>
</tr>
<tr>
<td>8. Cell</td>
<td>316</td>
<td>503</td>
</tr>
<tr>
<td>10. International Conference on Learning Representations</td>
<td>303</td>
<td>563</td>
</tr>
</tbody>
</table>

h5-index: largest number h such that h articles published in the last 5 years have at least h citations each.

[https://scholar.google.com/citations?view_op=top_venues&hl=en](https://scholar.google.com/citations?view_op=top_venues&hl=en)
## CVPR 2023 by the Numbers

Selecting a category below changes the paper list on the right.

### Top 10 overall by number of authors

<table>
<thead>
<tr>
<th>Category</th>
<th>Authors</th>
<th>Papers</th>
</tr>
</thead>
<tbody>
<tr>
<td>3D from multi-view and sensors</td>
<td>1,090</td>
<td>246</td>
</tr>
<tr>
<td>Image and video synthesis and generation</td>
<td>889</td>
<td>185</td>
</tr>
<tr>
<td>Humans: Face, body, pose, gesture, movement</td>
<td>813</td>
<td>166</td>
</tr>
<tr>
<td>Transfer, meta, low-shot, continual, or long-tail learning</td>
<td>688</td>
<td>153</td>
</tr>
<tr>
<td>Recognition: Categorization, detection, retrieval</td>
<td>673</td>
<td>139</td>
</tr>
<tr>
<td>Vision, language, and reasoning</td>
<td>631</td>
<td>118</td>
</tr>
<tr>
<td>Low-level vision</td>
<td>553</td>
<td>126</td>
</tr>
<tr>
<td>Segmentation, grouping and shape analysis</td>
<td>524</td>
<td>113</td>
</tr>
<tr>
<td>Deep learning architectures and techniques</td>
<td>485</td>
<td>92</td>
</tr>
<tr>
<td>Multi-modal learning</td>
<td>450</td>
<td>89</td>
</tr>
<tr>
<td>3D from single images</td>
<td>431</td>
<td>91</td>
</tr>
<tr>
<td>Medical and biological vision, cell microscopy</td>
<td>420</td>
<td>53</td>
</tr>
<tr>
<td>Video: Action and event understanding</td>
<td>373</td>
<td>63</td>
</tr>
<tr>
<td>Autonomous driving</td>
<td>359</td>
<td>69</td>
</tr>
<tr>
<td>Self-supervised or unsupervised representation learning</td>
<td>349</td>
<td>71</td>
</tr>
<tr>
<td>Datasets and evaluation</td>
<td>344</td>
<td>64</td>
</tr>
<tr>
<td>Scene analysis and understanding</td>
<td>276</td>
<td>64</td>
</tr>
<tr>
<td>Adversarial attack and defense</td>
<td>274</td>
<td>61</td>
</tr>
<tr>
<td>Efficient and scalable vision</td>
<td>252</td>
<td>48</td>
</tr>
<tr>
<td>Computational imaging</td>
<td>240</td>
<td>53</td>
</tr>
<tr>
<td>Video: Low-level analysis, motion, and tracking</td>
<td>215</td>
<td>46</td>
</tr>
<tr>
<td>Vision applications and systems</td>
<td>176</td>
<td>65</td>
</tr>
<tr>
<td>Vision + graphics</td>
<td>158</td>
<td>82</td>
</tr>
<tr>
<td>Robotics</td>
<td>141</td>
<td>23</td>
</tr>
<tr>
<td>Transparency, fairness, accountability, privacy, ethics in vision</td>
<td>129</td>
<td>80</td>
</tr>
<tr>
<td>Explainable computer vision</td>
<td>107</td>
<td>24</td>
</tr>
<tr>
<td>Embodied vision: Active agents, simulation</td>
<td>86</td>
<td>14</td>
</tr>
<tr>
<td>Document analysis and understanding</td>
<td>72</td>
<td>12</td>
</tr>
<tr>
<td>Machine learning (other than deep learning)</td>
<td>65</td>
<td>14</td>
</tr>
<tr>
<td>Physics-based vision and shape-from-X</td>
<td>55</td>
<td>12</td>
</tr>
</tbody>
</table>

### Papers

- **NeuMap: Neural Coordinate Mapping by Auto-Transdecoder for Camera Localization**
- **Object Pose Estimation with Statistical Guarantees: Conformal Keypoint Detection and Geometric Uncertainty Propagation**
- **NeuralUDF: Learning Unsigned Distance Fields for Multi-view Reconstruction of Surfaces with Arbitrary Topologies**
- **NEF: Neural Edge Fields for 3D Parametric Curve Reconstruction from Multi-view Images**
- **Looking Through the Glass: Neural Surface Reconstruction Against High Specular Reflections**
- **Multi-View Azimuth Stereo via Tangent Space Consistency**

[https://cvpr2023.thecvf.com/Conferences/2023/AcceptedPapers](https://cvpr2023.thecvf.com/Conferences/2023/AcceptedPapers)
Why do we care about Geometry?

Self-driving cars: navigation, collision avoidance

Robots: navigation, manipulation

Graphics & AR/VR: augment or generate images

Photogrammetry (architecture, surveys)

Pattern Recognition (web, medical imaging, etc)
Geometry is more useful now than ever!
Overview of Geometric Vision in CS131

Geometric Image Formation

The Pinhole Camera model + Calibration

Multi-view Geometry

Structure-from-Motion

Reference textbooks: Szeliski, Hartley & Zisserman to go deeper

Slides credits: Fei-Fei Li, JC Niebles, J. Wu, K. Kitani, S. Lazebnik, S. Seitz, D. Fouhey, J. Johnson
What will we learn today?

Why Geometric Vision Matters

Geometric Primitives in 2D & 3D

2D & 3D Transformations
General Advice / Observations

Fundamentals: need to (eventually) feel easy

Try to do the math in parallel live in class!

If not grokking this: practice later, ask on Ed, OH

Lots of good (hard?) exercises in Szeliski's book
What will we learn today?

Why Geometric Vision Matters

Geometric Primitives in 2D & 3D

2D & 3D Transformations
Images are 2D projections of the 3D world
Simplified Image Formation

Figure: R. Szeliski
Can we understand the 3D world from 2D images?
CV is an **ill-posed** inverse problem

2D Image

3D Scene

Graphics

Vision

Pixel Matrix

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>217</td>
<td>191</td>
<td>252</td>
<td>255</td>
<td>239</td>
</tr>
<tr>
<td>102</td>
<td>80</td>
<td>200</td>
<td>146</td>
<td>138</td>
</tr>
<tr>
<td>159</td>
<td>94</td>
<td>91</td>
<td>121</td>
<td>138</td>
</tr>
<tr>
<td>179</td>
<td>106</td>
<td>136</td>
<td>85</td>
<td>41</td>
</tr>
<tr>
<td>115</td>
<td>129</td>
<td>83</td>
<td>112</td>
<td>67</td>
</tr>
<tr>
<td>94</td>
<td>114</td>
<td>105</td>
<td>111</td>
<td>89</td>
</tr>
</tbody>
</table>

Objects

- Shape/Geometry
- Semantics

Material

- Motion
- 3D Pose

Slide credit: Andreas Geiger
Brief History of Geometric Vision

• 2020-: geometry + learning
• 2010s: deep learning
• 2000s: local features, birth of benchmarks
• 1990s: digital camera, 3D reconstruction
• 1980s: epipolar geometry (stereo) [Longuet-Higgins]
• ...

Brief History of Geometric Vision

• 1860s: first Computer Vision startup? [Willème]

Source: P. Sturm
Brief History of Geometric Vision

• 1860s: first Computer Vision startup? [Willème]
• 1850s: birth of photogrammetry [Laussedat]
• 1840s: panoramic photography

Source: P. Sturm
Brief History of Geometric Vision

• 1860s: first Computer Vision startup? [Willème]
• 1850s: birth of photogrammetry [Laussedat]
• 1840s: panoramic photography
• 1822-39: birth of photography [Niéepe, Daguerre]
• 1773: general 3-point pose estimation [Lagrange]
• 1715: basic intrinsic calibration (pre-photography!) [Taylor]
• 1700’s: topographic mapping from perspective drawings [Beaupré, Kappeler]

Source: P. Sturm
Brief History of Geometric Vision

• 15th century: start of mathematical treatment of 3D, first AR app?

Augmented reality invented by Filippo Brunelleschi (1377-1446)?

Tavoletta prospettica di Brunelleschi

Source: P. Sturm
Brief History of Geometric Vision

- 5th century BC: principles of pinhole camera, a.k.a. camera obscura
  - China: 5th century BC
  - Greece: 4th century BC
  - Egypt: 11th century
  - Throughout Europe: from 11th century onwards

First mention ...

Chinese philosopher Mozi (470 to 390 BC)

Greek philosopher Aristotle (384 to 322 BC)

Source: P. Sturm
What will we learn today?

Why Geometric Vision Matters

Geometric Primitives in 2D & 3D

2D & 3D Transformations
Points

2D points: \( \mathbf{x} = (x, y) \in \mathbb{R}^2 \) or column vector \( \mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix} \)

3D points: \( \mathbf{x} = (x, y, z) \in \mathbb{R}^3 \) (often noted \( \mathbf{X} \) or \( \mathbf{P} \))

Homogeneous coordinates: append a 1

\[ \bar{\mathbf{x}} = (x, y, 1) \quad \bar{\mathbf{x}} = (x, y, z, 1) \]

Why?
Everything is easier in Projective Space

2D Lines:
  Representation: $l = (a, b, c)$
  Equation: $ax + by + c = 0$
  In homogeneous coordinates: $\bar{x}^T l = 0$

General idea: homogenous coordinates unlock the full power of linear algebra!
Homogeneous coordinates in 2D

2D Projective Space: \( \mathcal{P}^2 = \mathbb{R}^3 - (0, 0, 0) \) (same story in 3D with \( \mathcal{P}^3 \))

- heterogeneous \( \rightarrow \) homogeneous
  \[
  \begin{bmatrix}
  x \\
  y
  \end{bmatrix}
  \Rightarrow
  \begin{bmatrix}
  x \\
  y \\
  1
  \end{bmatrix}
  \]

- homogeneous \( \rightarrow \) heterogeneous
  \[
  \begin{bmatrix}
  x \\
  y \\
  w
  \end{bmatrix}
  \Rightarrow
  \begin{bmatrix}
  x/w \\
  y/w
  \end{bmatrix}
  \]

- points differing only by scale are equivalent: \((x, y, w) \sim \lambda (x, y, w)\)

\[
\tilde{x} = (\tilde{x}, \tilde{y}, \tilde{w}) = \tilde{w}(x, y, 1) = \tilde{w}\tilde{x}.
\]
Everything is easier in Projective Space

2D Lines:
\[ \tilde{x}^T l = 0, \forall \tilde{x} = (x, y, w) \in P^2 \]
\[ l = (\hat{n}_x, \hat{n}_y, d) = (\hat{n}, d) \text{ with } \|\hat{n}\| = 1 \]

3D planes: same!
\[ \tilde{x}^T m = 0, \forall \tilde{x} = (x, y, z, w) \in P^3 \]
\[ m = (\hat{n}_x, \hat{n}_y, \hat{n}_z, d) = (\hat{n}, d) \text{ with } \|\hat{n}\| = 1 \]
**Lines in 3D**

**Two-point parametrization:**

\[ \mathbf{r} = (1 - \lambda)\mathbf{p} + \lambda\mathbf{q} \quad \tilde{\mathbf{r}} = \mu\tilde{\mathbf{p}} + \lambda\tilde{\mathbf{q}} \]

**Two-plane parametrization:**

coordinates \((x_0, y_0)\) & \((x_1, y_1)\) of intersection with planes at \(z = 0, 1\) (or other planes)
Cross-product quick reminder

\[
\mathbf{a} \times \mathbf{b} = \|\mathbf{a}\| \|\mathbf{b}\| \sin(\theta) \mathbf{n}
\]

\[
\mathbf{a} \times \mathbf{b} = [\mathbf{a}] \times \mathbf{b} = \\
\begin{bmatrix}
0 & -a_3 & a_2 \\
a_3 & 0 & -a_1 \\
-a_2 & a_1 & 0 \\
\end{bmatrix} \\
\begin{bmatrix}
b_1 \\
b_2 \\
b_3 \\
\end{bmatrix}
\]
Benefits of Homogeneous Coordinates

• Line – Point duality:
  • line between two 2D points: \( \tilde{l} = \tilde{x}_1 \times \tilde{x}_2 \)
  • intersection of two 2D lines: \( \tilde{x} = \tilde{l}_1 \times \tilde{l}_2 \)

• Representation of Infinity:
  • points at infinity: \((x, y, 0)\); line at infinity: \((0,0,1)\)

• Parallel & vertical lines are easy (take-home: intersect //)

• Makes 2D & 3D transformations linear!
Questions?
What will we learn today?

Why Geometric Vision Matters

Geometric Primitives in 2D & 3D

2D & 3D Transformations
The camera as a coordinate transformation

A camera is a mapping from:

the 3D world
to:

a 2D image

Source: K. Kitani
Cameras and objects can move!

**Figure 2.12**  A point is projected into two images: (a) relationship between the 3D point coordinate \((X, Y, Z, 1)\) and the 2D projected point \((x, y, 1, d)\); (b) planar homography induced by points all lying on a common plane \(\hat{n}_0 \cdot p + c_0 = 0\).
2D Transformations Zoo

Figure: R. Szeliski
**Transformation = Matrix Multiplication**

<table>
<thead>
<tr>
<th>Transformation</th>
<th>Matrix Representation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scale</td>
<td>$M = \begin{bmatrix} s_x &amp; 0 \ 0 &amp; s_y \end{bmatrix}$</td>
</tr>
<tr>
<td>Flip across y</td>
<td>$M = \begin{bmatrix} -1 &amp; 0 \ 0 &amp; 1 \end{bmatrix}$</td>
</tr>
<tr>
<td>Rotate</td>
<td>$M = \begin{bmatrix} \cos \theta &amp; -\sin \theta \ \sin \theta &amp; \cos \theta \end{bmatrix}$</td>
</tr>
<tr>
<td>Flip across origin</td>
<td>$M = \begin{bmatrix} -1 &amp; 0 \ 0 &amp; -1 \end{bmatrix}$</td>
</tr>
<tr>
<td>Shear</td>
<td>$M = \begin{bmatrix} 1 &amp; s_x \ s_y &amp; 1 \end{bmatrix}$</td>
</tr>
<tr>
<td>Identity</td>
<td>$M = \begin{bmatrix} 1 &amp; 0 \ 0 &amp; 1 \end{bmatrix}$</td>
</tr>
</tbody>
</table>
Scaling

\[
\begin{bmatrix}
  s_x & 0 \\
  0 & s_y \\
\end{bmatrix}
\begin{bmatrix}
  x \\
  y \\
\end{bmatrix}
=
\begin{bmatrix}
  s_xx \\
  s_yy \\
\end{bmatrix}
\]
Rotation

$x' = x \cos \theta - y \sin \theta$
$y' = x \sin \theta + y \cos \theta$

or in matrix form:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

Rotation matrix:
- Inverse is transpose
- Orthonormal

$R \cdot R^T = R^T \cdot R = I$
$\det(R) = 1$

Slide: K. Kitani
2D Translation

\[ x' = x + t_x \]
\[ y' = y + t_y \]

As a matrix?

Slide: JC. Niebles
2D Translation with homogeneous coordinates

\[ p = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \]

\[ t = \begin{bmatrix} t_x \\ t_y \\ 1 \end{bmatrix} \rightarrow \begin{bmatrix} t_x \\ t_y \\ 1 \end{bmatrix} \]

\[ p' = Tp \]

\[ p' \rightarrow \begin{bmatrix} x + t_x \\ y + t_y \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} I & t \\ 0 & 1 \end{bmatrix} p = Tp \]
2D Transformations with homogeneous coordinates

No change
\[
\begin{bmatrix}
1 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 1
\end{bmatrix}
\]

Translate
\[
\begin{bmatrix}
1 & 0 & X \\
0 & 1 & Y \\
0 & 0 & 1
\end{bmatrix}
\]

Scale about origin
\[
\begin{bmatrix}
W & 0 & 0 \\
0 & H & 0 \\
0 & 0 & 1
\end{bmatrix}
\]

Rotate about origin
\[
\begin{bmatrix}
\cos \theta & -\sin \theta & 0 \\
\sin \theta & \cos \theta & 0 \\
0 & 0 & 1
\end{bmatrix}
\]

Shear in x direction
\[
\begin{bmatrix}
1 & \tan \phi & 0 \\
0 & 1 & 0 \\
0 & 0 & 1
\end{bmatrix}
\]

Shear in y direction
\[
\begin{bmatrix}
1 & 0 & 0 \\
0 & \tan \psi & 1 \\
0 & 0 & 1
\end{bmatrix}
\]

Figure: Wikipedia
Questions?
2D Transformations Zoo

Figure: R. Szeliski
Euclidean / Rigid

Euclidean (rigid): rotation + translation

SE(2): Special Euclidean group
Important in robotics: describes poses on plane

\[
\begin{bmatrix}
\cos \theta & -\sin \theta & t_x \\
\sin \theta & \cos \theta & t_y \\
0 & 0 & 1
\end{bmatrix}
\]

How many degrees of freedom?
Similarity

Similarity:
Scaling
+ rotation
+ translation

\[
\begin{bmatrix}
a & -b & t_x \\
b & a & t_y \\
0 & 0 & 1
\end{bmatrix}
\]
Affine transformations are combinations of

• arbitrary (4-DOF) linear transformations; and

• translations

\[
\begin{bmatrix}
    x' \\
    y' \\
    w'
\end{bmatrix} = \begin{bmatrix}
    a & b & c \\
    d & e & f \\
    0 & 0 & 1
\end{bmatrix} \begin{bmatrix}
    x \\
    y \\
    w
\end{bmatrix}
\]

Properties of affine transformations:

• origin does not necessarily map to origin

• lines map to lines

• parallel lines map to parallel lines

• ratios are preserved

Source: K. Kitani
Projective transformation (homography)

Projective transformations are combinations of
- affine transformations; and
- projective warps

Properties of projective transformations:
- origin does not necessarily map to origin
- lines map to lines
- parallel lines do not necessarily map to parallel lines
- ratios are not necessarily preserved

\[
\begin{bmatrix}
    x' \\
    y' \\
    w'
\end{bmatrix} = \begin{bmatrix}
    a & b & c \\
    d & e & f \\
    g & h & i
\end{bmatrix} \begin{bmatrix}
    x \\
    y \\
    w
\end{bmatrix}
\]

How many degrees of freedom?

Source: K. Kitani
Projective transformation (homography)

Projective transformations are combinations of
• affine transformations; and
• projective warps

Properties of projective transformations:
• origin does not necessarily map to origin
• lines map to lines
• parallel lines do not necessarily map to parallel lines
• ratios are not necessarily preserved

\[
\begin{bmatrix}
x' \\
y' \\
w'
\end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix}
\]

8 DOF: vectors (and therefore matrices) are defined up to scale

Source: K. Kitani
Questions?
Composing Transformations

Transformations = Matrices => Composition by Multiplication!

\[ p' = R_2 R_1 S p \]

In the example above, the result is equivalent to

\[ p' = R_2 (R_1 (S p)) \]

Equivalent to multiply the matrices into single transformation matrix:

\[ p' = (R_2 R_1 S) p \]

Order Matters! Transformations from right to left.
Scaling & Translating != Translating & Scaling

\[ p'' = TSp = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & t_x \\ 0 & s_y & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} s_xx + t_x \\ s_yy + t_y \\ 1 \end{bmatrix} \]

\[ p''' = STp = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & s_xt_x \\ 0 & s_y & s_yt_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} s_xx + s_xt_x \\ s_yy + s_yt_y \\ 1 \end{bmatrix} \]
Scaling + Rotation + Translation

\[ p' = (T R S) p \]

\[
p' = TRSp = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}
\]

\[
= \begin{bmatrix} \cos \theta & -\sin \theta & t_x \\ \sin \theta & \cos \theta & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}
\]

\[
= \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} S & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} RS & t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}
\]

This is the form of the general-purpose transformation matrix.
### 2D Transforms = Matrix Multiplication

<table>
<thead>
<tr>
<th>Transformation</th>
<th>Matrix</th>
<th># DoF</th>
<th>Preserves</th>
<th>Icon</th>
</tr>
</thead>
<tbody>
<tr>
<td>translation</td>
<td>$\begin{bmatrix} I &amp; t \end{bmatrix}_{2\times3}$</td>
<td>2</td>
<td>orientation</td>
<td></td>
</tr>
<tr>
<td>rigid (Euclidean)</td>
<td>$\begin{bmatrix} R &amp; t \end{bmatrix}_{2\times3}$</td>
<td>3</td>
<td>lengths</td>
<td></td>
</tr>
<tr>
<td>similarity</td>
<td>$\begin{bmatrix} sR &amp; t \end{bmatrix}_{2\times3}$</td>
<td>4</td>
<td>angles</td>
<td></td>
</tr>
<tr>
<td>affine</td>
<td>$\begin{bmatrix} A \end{bmatrix}_{2\times3}$</td>
<td>6</td>
<td>parallelism</td>
<td></td>
</tr>
<tr>
<td>projective</td>
<td>$\begin{bmatrix} \tilde{H} \end{bmatrix}_{3\times3}$</td>
<td>8</td>
<td>straight lines</td>
<td></td>
</tr>
</tbody>
</table>

**Table 2.1**  
Hierarchy of 2D coordinate transformations, listing the transformation name, its matrix form, the number of degrees of freedom, what geometric properties it preserves, and a mnemonic icon. Each transformation also preserves the properties listed in the rows below it, i.e., similarity preserves not only angles but also parallelism and straight lines. The $2 \times 3$ matrices are extended with a third $[0^T \ 1]$ row to form a full $3 \times 3$ matrix for homogeneous coordinate transformations.
Questions?
### 3D Transforms = Matrix Multiplication

<table>
<thead>
<tr>
<th>Transformation</th>
<th>Matrix</th>
<th># DoF</th>
<th>Preserves</th>
<th>Icon</th>
</tr>
</thead>
<tbody>
<tr>
<td>translation</td>
<td>( \begin{bmatrix} I &amp; t \end{bmatrix}_{3\times4} )</td>
<td>3</td>
<td>orientation</td>
<td>□</td>
</tr>
<tr>
<td>rigid (Euclidean)</td>
<td>( \begin{bmatrix} R &amp; t \end{bmatrix}_{3\times4} )</td>
<td>6</td>
<td>lengths</td>
<td>◊</td>
</tr>
<tr>
<td>similarity</td>
<td>( \begin{bmatrix} sR &amp; t \end{bmatrix}_{3\times4} )</td>
<td>7</td>
<td>angles</td>
<td>◊</td>
</tr>
<tr>
<td>affine</td>
<td>( \begin{bmatrix} A \end{bmatrix}_{3\times4} )</td>
<td>12</td>
<td>parallelism</td>
<td>□</td>
</tr>
<tr>
<td>projective</td>
<td>( \begin{bmatrix} \hat{H} \end{bmatrix}_{4\times4} )</td>
<td>15</td>
<td>straight lines</td>
<td>□</td>
</tr>
</tbody>
</table>

Table 2.2  Hierarchy of 3D coordinate transformations. Each transformation also preserves the properties listed in the rows below it, i.e., similarity preserves not only angles but also parallelism and straight lines. The \( 3 \times 4 \) matrices are extended with a fourth \( [0^T \ 1] \) row to form a full \( 4 \times 4 \) matrix for homogeneous coordinate transformations. The mnemonic icons are drawn in 2D but are meant to suggest transformations occurring in a full 3D cube.

Figure: R. Szeliski
3D Rotations: \( \text{SO}(3) \) representations

**Euler Angles:** yaw, pitch, roll \((\alpha, \beta, \gamma)\)
→ compose \( R(\gamma)R(\beta)R(\alpha) \) (order, axes!)

**Axis-angle:** \((\hat{n}, \theta)\) or \( \omega = \theta \hat{n} \)
→ matrix via Rodrigues formula (simple for small \( \theta \))
\[
R(\hat{n}, \theta) = I + \sin \theta [\hat{n}]_\times + (1 - \cos \theta) [\hat{n}]_\times^2 \approx I + [\theta \hat{n}]_\times
\]

**Unit Quaternions:** \( q = (x, y, z, w) = (\sin \frac{\theta}{2} \hat{n}, \cos \frac{\theta}{2}) \), \( ||q|| = 1 \)
→ continuous, nice algebraic properties, matrix via Rodrigues

\[
R(q) = \begin{bmatrix}
1 - 2(y^2 + z^2) & 2(xy - zw) & 2(xz + yw) \\
2(xy + zw) & 1 - 2(x^2 + z^2) & 2(yz - xw) \\
2(xz - yw) & 2(yz + xw) & 1 - 2(x^2 + y^2)
\end{bmatrix}
\]

See Szeliski 2.1.3 for more details
Questions?
What did we learn today?

Geometry is essential to Computer Vision!

Geometric Primitives in 2D & 3D

homogeneous coordinates, points, lines, and planes in 2D & 3D

2D & 3D Transformations

scaling, translation, rotation, rigid, similarity, affine, homography

Next Lecture: putting this in “perspective”…
Appendix
Intersecting Parallel Lines

\[ \tilde{l}_1 = (a_1, b_1, c_1) \]

\[ \tilde{l}_2 = (a_2, b_2, c_2) \]
Intersecting Parallel Lines

\[ \tilde{x} = \tilde{l}_1 \times \tilde{l}_2 \]

\[ \tilde{x} \sim (b_1, -a_1, 0) \]

\[ \tilde{l}_1 = (a_1, b_1, c_1) \]

\[ \tilde{l}_2 = (a_2, b_2, c_2) \]

\[ -\frac{a_1}{b_1} = -\frac{a_2}{b_2} \]

\[ (a_2, b_2) = w(a_1, b_1) \]
2D planar transformations

Polar coordinates...
\[ x = r \cos (\varphi) \]
\[ y = r \sin (\varphi) \]
\[ x' = r \cos (\varphi + \theta) \]
\[ y' = r \sin (\varphi + \theta) \]

Trigonometric Identity...
\[ x' = r \cos(\varphi) \cos(\theta) - r \sin(\varphi) \sin(\theta) \]
\[ y' = r \sin(\varphi) \cos(\theta) + r \cos(\varphi) \sin(\theta) \]

Substitute...
\[ x' = x \cos(\theta) - y \sin(\theta) \]
\[ y' = x \sin(\theta) + y \cos(\theta) \]