**CS4620/5620:** Pipeline and Transformations

Professor: Kavita Bala

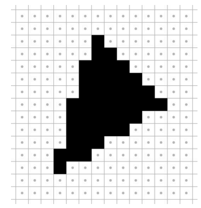



## Object-order vs. Image-order

- Object-order

```
for each triangle t {
  find pixels covered by t
  c(x,y) = shaded result
}
```

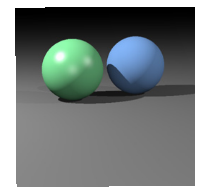
- Hardware pipeline



- Image-order

```
for each pixel p=(x,y) {
  intersect ray through p with scene
  c(x,y) = shade (visible point)
}
```

- Ray tracers






## Math review

- **Read:**
  - **Chapter 2: Miscellaneous Math**
  - **Chapter 5: Linear Algebra**
- Vectors and points
- Vector operations
  - addition
  - scalar product
- More products
  - dot product
  - cross product
- Bases and orthogonality




## Geometry of 2D linear trans.

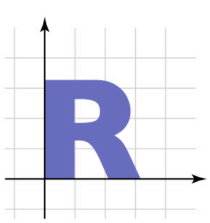
- 2x2 matrices have simple geometric interpretations
  - uniform scale
  - non-uniform scale
  - reflection
  - shear
  - rotation

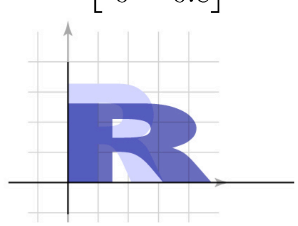



## Linear transformation gallery

- Nonuniform scale $\begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} s_x x \\ s_y y \end{bmatrix}$

$$\begin{bmatrix} 1.5 & 0 \\ 0 & 0.8 \end{bmatrix}$$



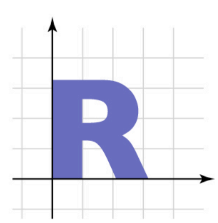


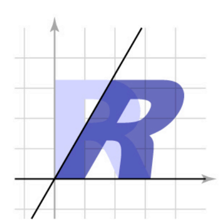



## Linear transformation gallery

- Shear $\begin{bmatrix} 1 & a \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x + ay \\ y \end{bmatrix}$

$$\begin{bmatrix} 1 & 0.5 \\ 0 & 1 \end{bmatrix}$$

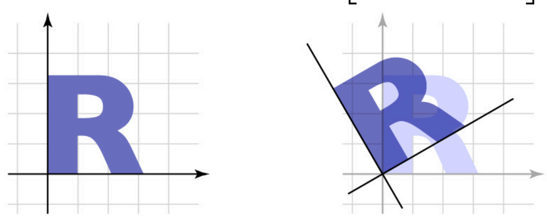




## Linear transformation gallery

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

$$\begin{bmatrix} 0.866 & -.05 \\ 0.5 & 0.866 \end{bmatrix}$$





## Linear transformations

- One way to define a transformation is by matrix multiplication:
  $$T(\mathbf{v}) = M\mathbf{v}$$
- Such transformations are *linear*, which is to say:
  $$T(a\mathbf{u} + \mathbf{v}) = aT(\mathbf{u}) + T(\mathbf{v})$$
  - (and in fact all linear transformations can be written this way)
- Is translation linear?



## Composing transformations

- Want to move an object, then move it some more
  - $\mathbf{p} \to T(\mathbf{p}) \to S(T(\mathbf{p})) = (S \circ T)(\mathbf{p})$
- We need to represent *S* o *T* (“S compose T”)
  - and would like to use the same representation as for *S* and *T*
- Translation easy
  - $T(\mathbf{p}) = \mathbf{p} + \mathbf{u}_T; S(\mathbf{p}) = \mathbf{p} + \mathbf{u}_S$
    $(S \circ T)(\mathbf{p}) = \mathbf{p} + (\mathbf{u}_T + \mathbf{u}_S)$
- Translation by $\mathbf{u}_T$ then by $\mathbf{u}_S$ is translation by $\mathbf{u}_T + \mathbf{u}_S$
  - commutative!



## Composing transformations

- Linear transformations also straightforward
  - $T(\mathbf{p}) = M_T\mathbf{p}; S(\mathbf{p}) = M_S\mathbf{p}$
    $(S \circ T)(\mathbf{p}) = M_S M_T \mathbf{p}$
- Transforming first by $M_T$ then by $M_S$ is the same as transforming by $M_S M_T$
  - only sometimes commutative
    - e.g. rotations & uniform scales
    - e.g. non-uniform scales w/o rotation
  - Note $M_S M_T$, or *S* o *T*, is *T* first, then *S*



## Combining linear with translation

- Need to use both in single framework
- Can represent arbitrary seq. as $T(\mathbf{p}) = M\mathbf{p} + \mathbf{u}$
  - $T(\mathbf{p}) = M_T\mathbf{p} + \mathbf{u}_T$
  - $S(\mathbf{p}) = M_S\mathbf{p} + \mathbf{u}_S$
  - $(S \circ T)(\mathbf{p}) = M_S(M_T\mathbf{p} + \mathbf{u}_T) + \mathbf{u}_S$
    $= (M_S M_T)\mathbf{p} + (M_S\mathbf{u}_T + \mathbf{u}_S)$
  - e.g. $S(T(0)) = S(\mathbf{u}_T)$
- Transforming by $M_T$ and $\mathbf{u}_T$, then by $M_S$ and $\mathbf{u}_S$, is the same as transforming by $M_S M_T$ and $\mathbf{u}_S + M_S\mathbf{u}_T$
  - This will work but is a little awkward



## Homogeneous coordinates

- A trick for representing the foregoing more elegantly
- Extra component *w* for vectors, extra row/column for matrices
  - for affine, can always keep *w* = 1
- Represent linear transformations with dummy extra row and column

$$\begin{bmatrix} a & b & 0 \\ c & d & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} ax + by \\ cx + dy \\ 1 \end{bmatrix}$$

## Homogeneous coordinates

- Represent translation using the extra column

$$\begin{bmatrix} 1 & 0 & t \\ 0 & 1 & s \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} x+t \\ y+s \\ 1 \end{bmatrix}$$

- What kind of transform is this?



## Homogeneous coordinates

- Composition just works, by 3x3 matrix multiplication

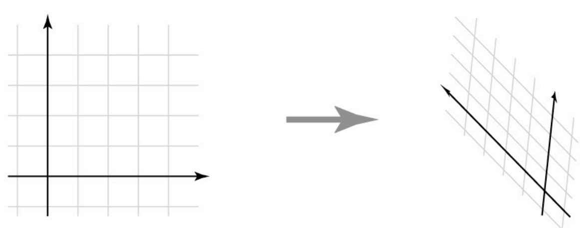
$$\begin{bmatrix} M_S & \mathbf{u}_S \\ 0 & 1 \end{bmatrix} \begin{bmatrix} M_T & \mathbf{u}_T \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{p} \\ 1 \end{bmatrix} = \begin{bmatrix} (M_S M_T)\mathbf{p} + (M_S \mathbf{u}_T + \mathbf{u}_S) \\ 1 \end{bmatrix}$$

- This is exactly the same as carrying around *M* and **u**
  - but cleaner
  - and generalizes in useful ways as we'll see later



## Affine transformations

- The set of transformations we are interested in is known as the "affine" transformations
  - straight lines preserved; parallel lines preserved
  - ratios of lengths along lines preserved (midpoints preserved)

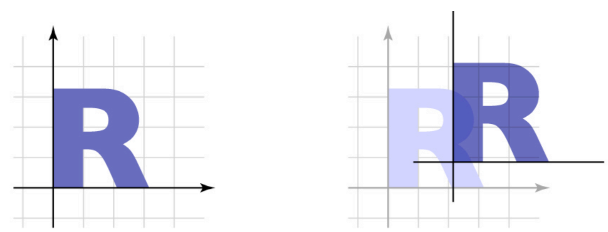




## Affine transformation gallery

- Translation

$$\begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \quad \begin{bmatrix} 1 & 0 & 2.15 \\ 0 & 1 & 0.85 \\ 0 & 0 & 1 \end{bmatrix}$$

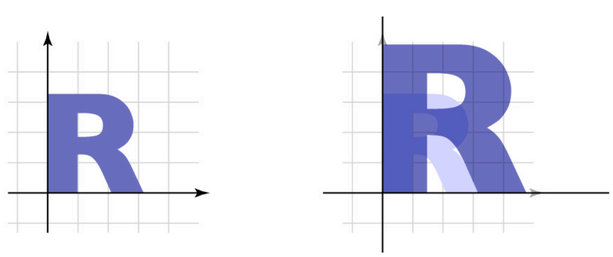




## Affine transformation gallery

- Uniform scale

$$\begin{bmatrix} s & 0 & 0 \\ 0 & s & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \begin{bmatrix} 1.5 & 0 & 0 \\ 0 & 1.5 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

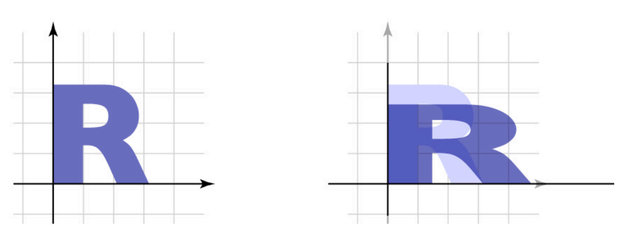




## Affine transformation gallery

- Nonuniform scale

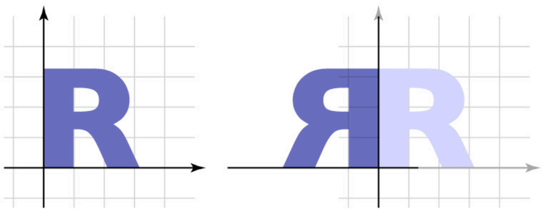
$$\begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \begin{bmatrix} 1.5 & 0 & 0 \\ 0 & 0.8 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$



## Affine transformation gallery

- Reflection
  - can consider it a special case of nonuniform scale

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

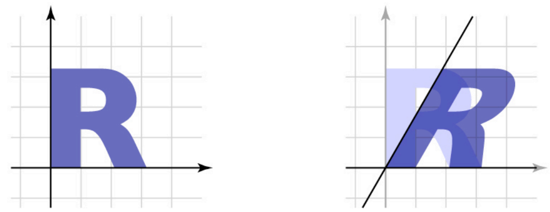




## Affine transformation gallery

- Shear

$$\begin{bmatrix} 1 & a & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \begin{bmatrix} 1 & 0.5 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

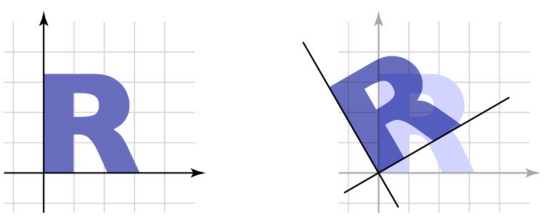




## Affine transformation gallery

- Rotation

$$\begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0.866 & -0.5 & 0 \\ 0.5 & 0.866 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$





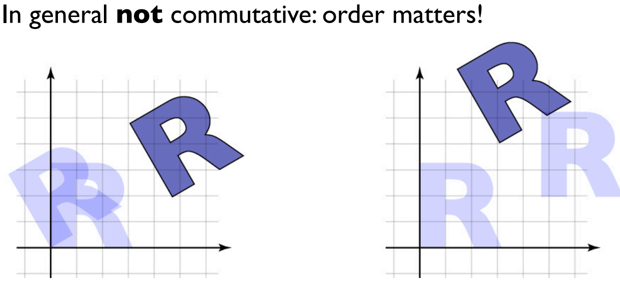
## General affine transformations

- The previous slides showed "canonical" examples of the types of affine transformations
- Generally, transformations contain elements of multiple types
  - often define them as products of canonical transforms
  - sometimes work with their properties more directly



## Composite affine transformations

- In general **not** commutative: order matters!



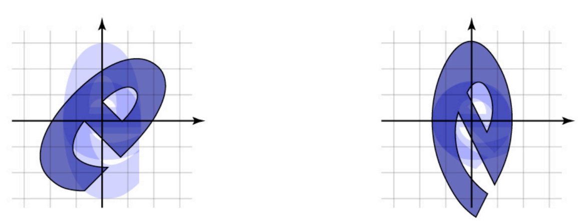
rotate, then translate

translate, then rotate



## Composite affine transformations

- Another example



scale, then rotate

rotate, then scale

## Rigid motions

- A transform made up of only translation and rotation is a *rigid motion* or a *rigid body transformation*
- The linear part is an orthonormal matrix

$$R = \begin{bmatrix} Q & \mathbf{u} \\ 0 & 1 \end{bmatrix}$$

- Inverse of orthonormal matrix is transpose
  – so inverse of rigid motion is easy:

$$R^{-1}R = \begin{bmatrix} Q^T & -Q^T\mathbf{u} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} Q & \mathbf{u} \\ 0 & 1 \end{bmatrix}$$





## Transforming points and vectors

- Recall distinction between points vs. vectors
  – vectors are just offsets (differences between points)
  – points have a location
    • represented by vector offset from a fixed origin
- Points and vectors transform differently
  – points respond to translation; vectors do not

$$\begin{aligned} \mathbf{v} &= \mathbf{p} - \mathbf{q} \\ T(\mathbf{x}) &= M\mathbf{x} + \mathbf{t} \\ T(\mathbf{p} - \mathbf{q}) &= M\mathbf{p} + \mathbf{t} - (M\mathbf{q} + \mathbf{t}) \\ &= M(\mathbf{p} - \mathbf{q}) + (\mathbf{t} - \mathbf{t}) = M\mathbf{v} \end{aligned}$$





## Transforming points and vectors

- Homogeneous coords. let us exclude translation
  – just put 0 rather than 1 in the last place

$$\begin{bmatrix} M & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix} \begin{bmatrix} \mathbf{p} \\ 1 \end{bmatrix} = \begin{bmatrix} M\mathbf{p} + \mathbf{t} \\ 1 \end{bmatrix} \quad \begin{bmatrix} M & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix} \begin{bmatrix} \mathbf{v} \\ 0 \end{bmatrix} = \begin{bmatrix} M\mathbf{v} \\ 0 \end{bmatrix}$$

  – and note that subtracting two points cancels the extra coordinate, resulting in a vector!
- Preview: projective transformations
  – what's really going on with this last coordinate?
  – think of $R^2$ embedded in $R^3$: all affine xfs. preserve z=1 plane
  – could have other transforms; project back to z=1