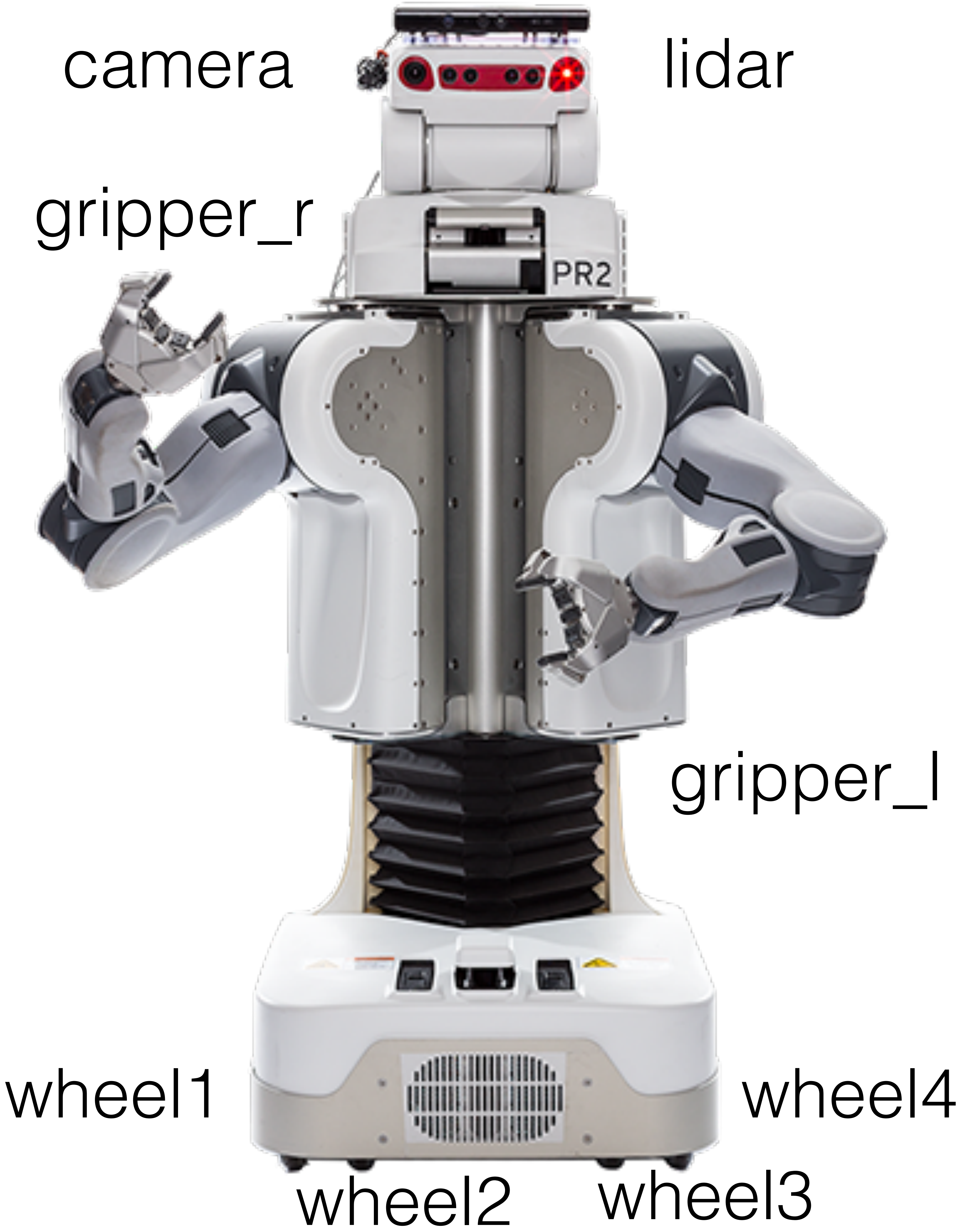
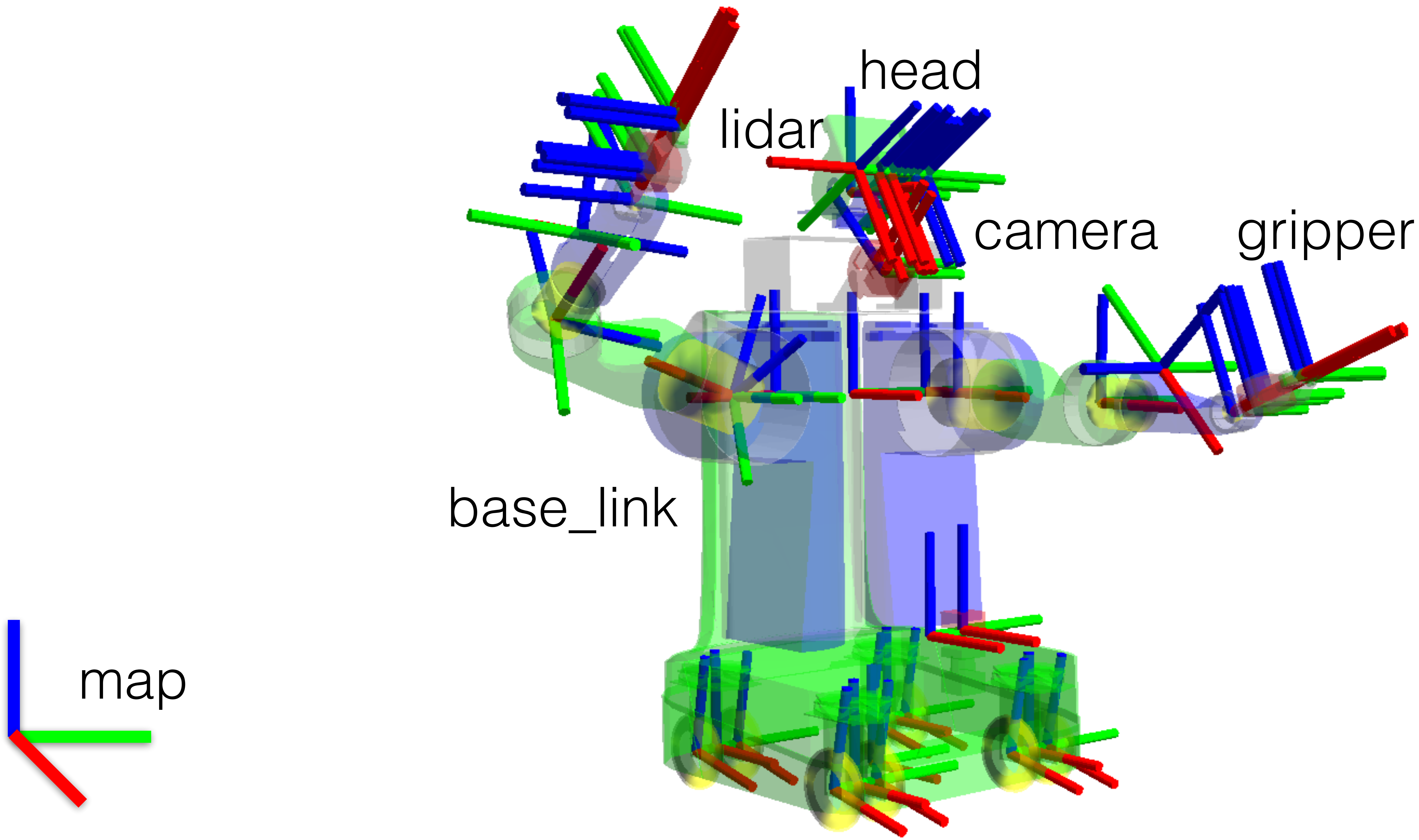
Transformation frames and lidar

Karel Zimmermann

Why transformation among coordinate frames are important?



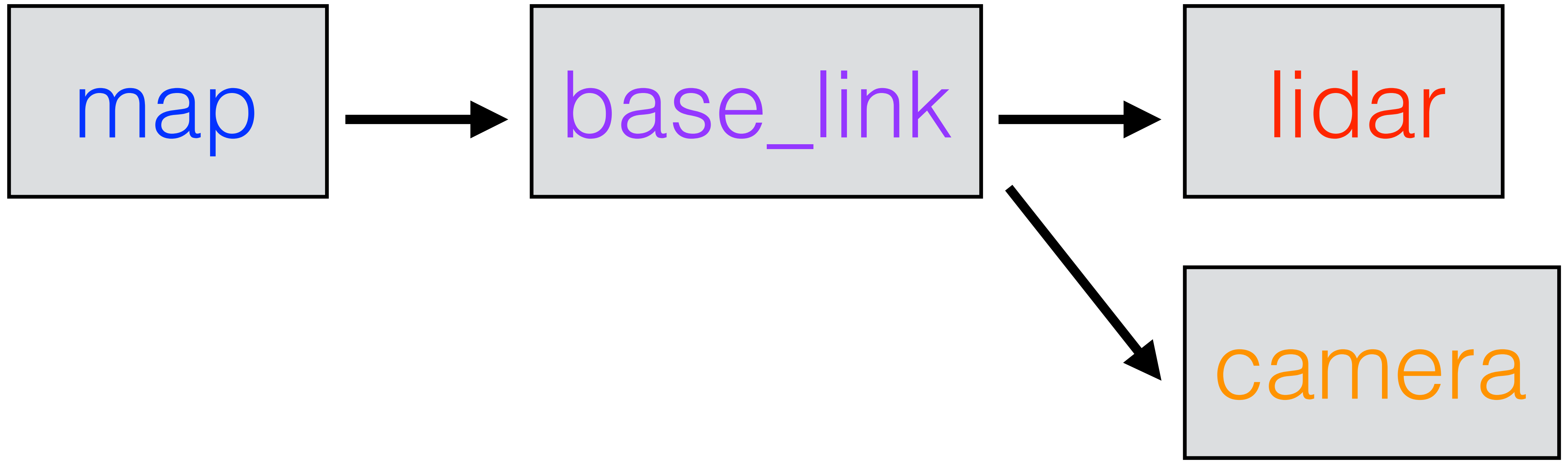
- Each coordinate frame define 3D Euclidean space.
- It is uniquely determined by its name.




File View Plugins Help
Displays
.Global Options
Background Color (0,0,0)
Fixed Frame /map
Target Frame /base_link
01. TF (TF)
02. Map (Map)
03. Laser Scan (Laser Scan)
Add
Remove
Target Frame

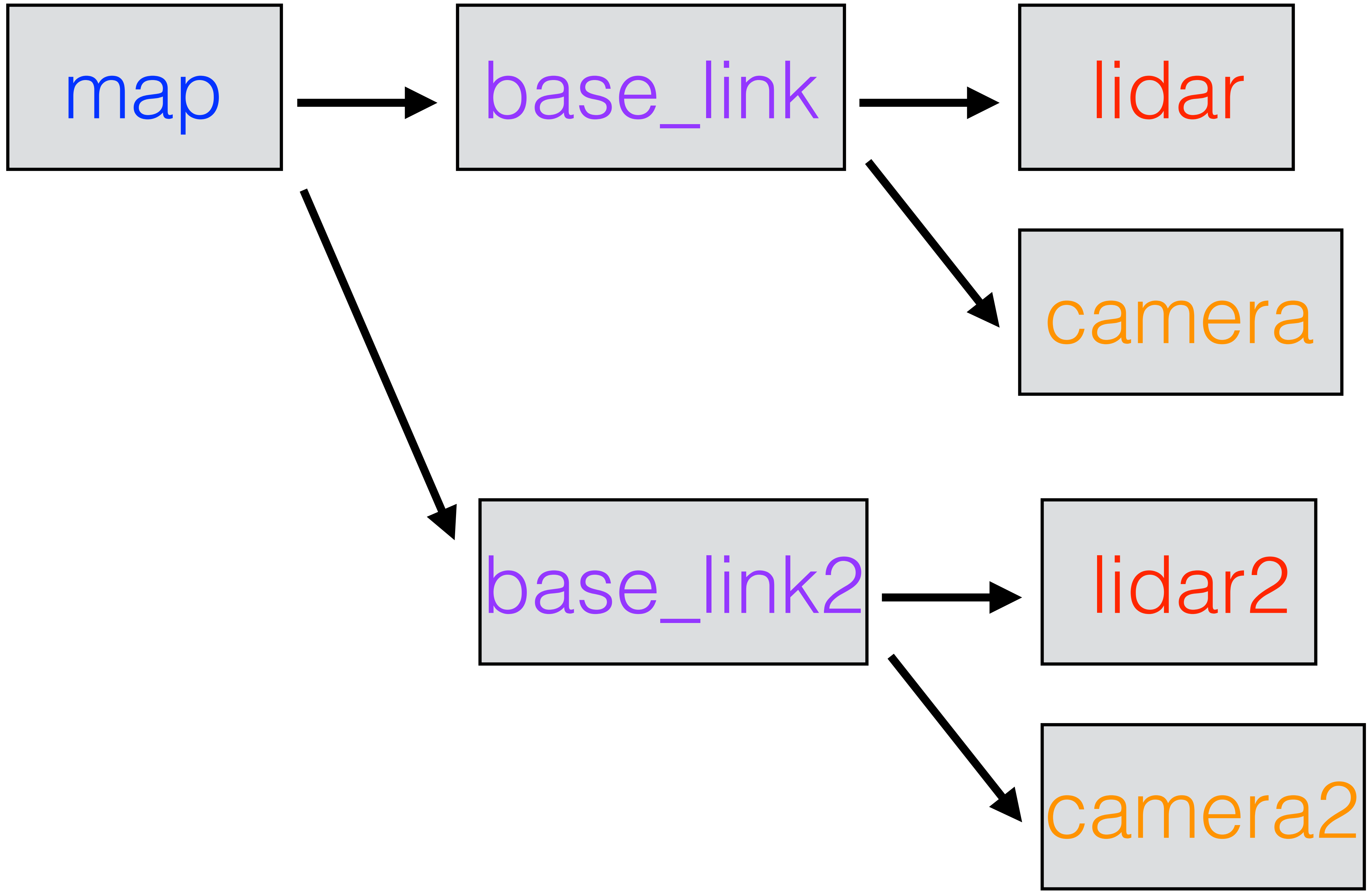
Coordinate frames & ROS

- Coordinate frames in ROS form a transformation tree: `tf`



Coordinate frames & ROS

- Transformation tree for two robots allow to estimate mutual position and use measurement from other robot.



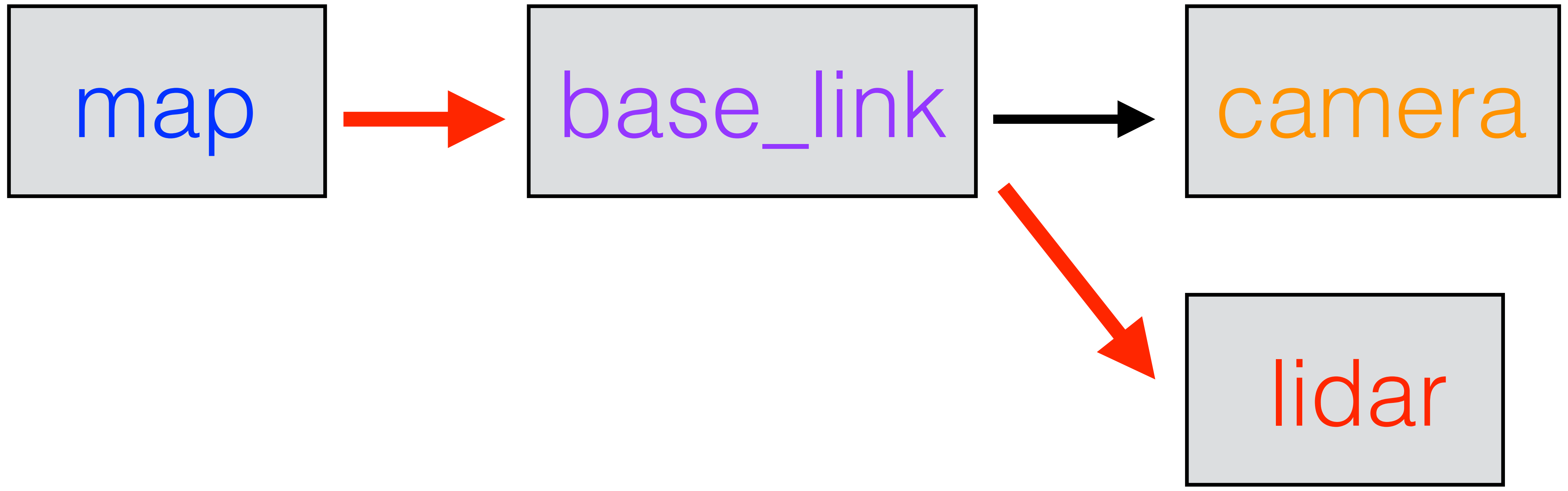
Listening static transformation between two c.f. in ROS ROS:

(1) in your own node:

```
# initialize listener (10 sec buffer)
buffer = tf2_ros.Buffer()
listener = tf2_ros.TransformListener(buffer)

# estimate transformation from lidar to map
transform = buffer.lookup_transform(‘lidar’,'map',rospy.Time())
```

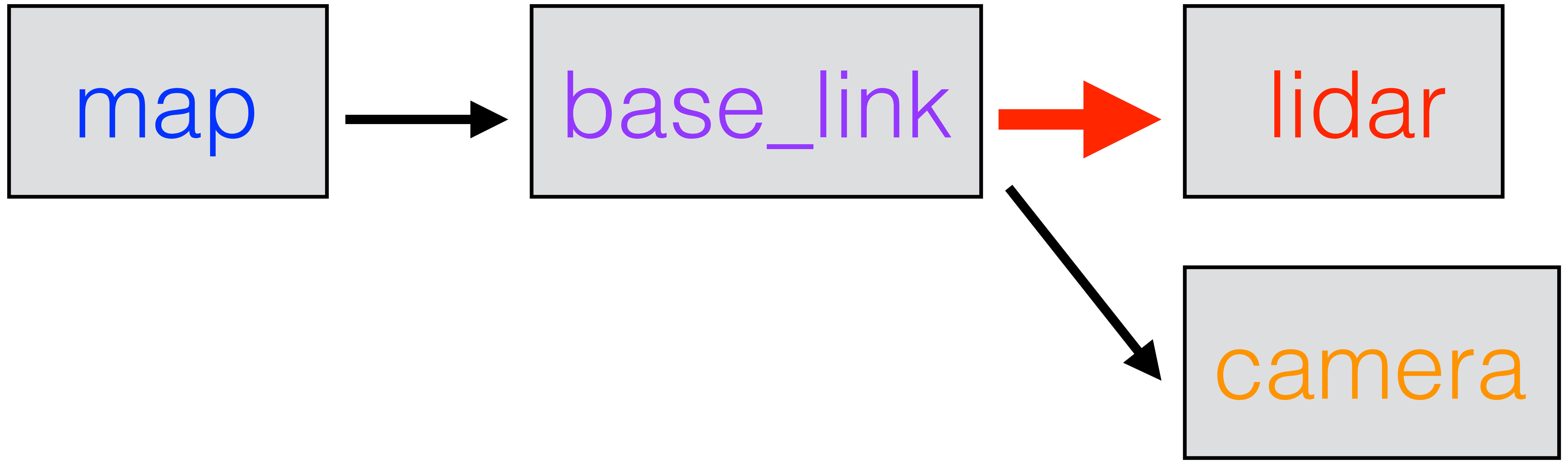
(2) In the terminal:

```
$ rosrun tf tf_echo /map /lidar
At time 1263248513.809
- Translation: [2.398, 6.783, 0.000]
- Rotation: in Quaternion [0.000, 0.000, -0.707, 0.707]
```



Coordinate frames & ROS

- Coordinate frames in ROS form a transformation tree: `tf`



- In order to let ROS to keep the `tf` up-to-date
  => **parent-child transformations** between all
  coordinate frames need to be published by your nodes

Broadcasting static transformation between two c.f. in ROS

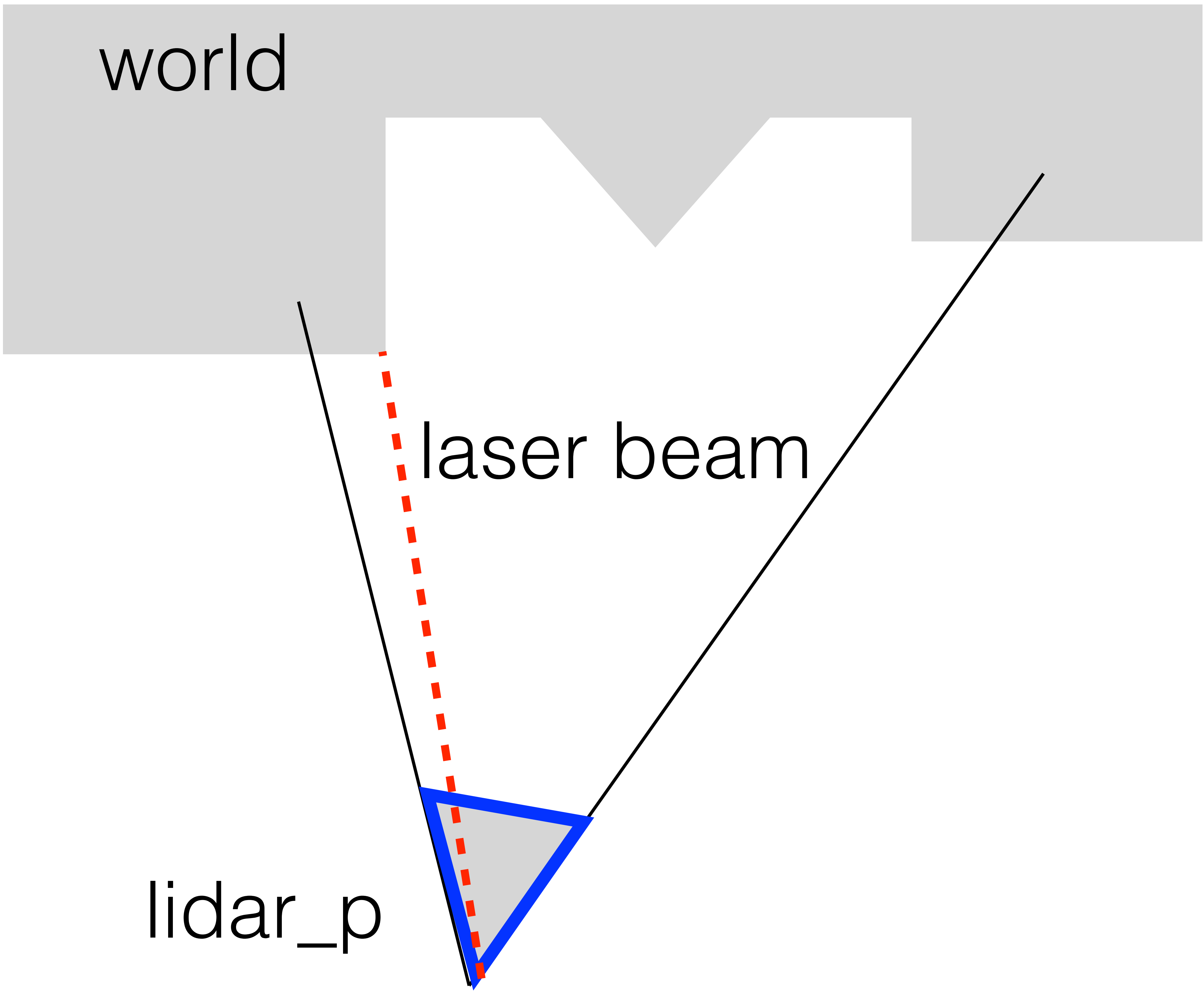
```
broadcaster = tf2_ros.StaticTransformBroadcaster()
transform = geometry_msgs.msg.TransformStamped()
# estimate R,t (e.g. measure or compute)
# … "TOPIC OF FOLLOWING TWO LECTURES" => R,t
# convert rotation matrix into quaternion
q = mat2quat(R)
# fill-in transform between coordinate frames (q,t)
transform.translation.x = t[0]
transform.translation.y = t[1]
transform.translation.z = t[2]
transform.rotation.x = q[0]
transform.rotation.y = q[1]
transform.rotation.z = q[2]
transform.rotation.w = q[3]
transform.header.stamp = rospy.Time.now()
transform.header.frame_id = "base_link"
transform.child_frame_id = "lidar"

# publish transform between coordinate frames (q,t)
broadcaster.sendTransform(transform)
```

Outline

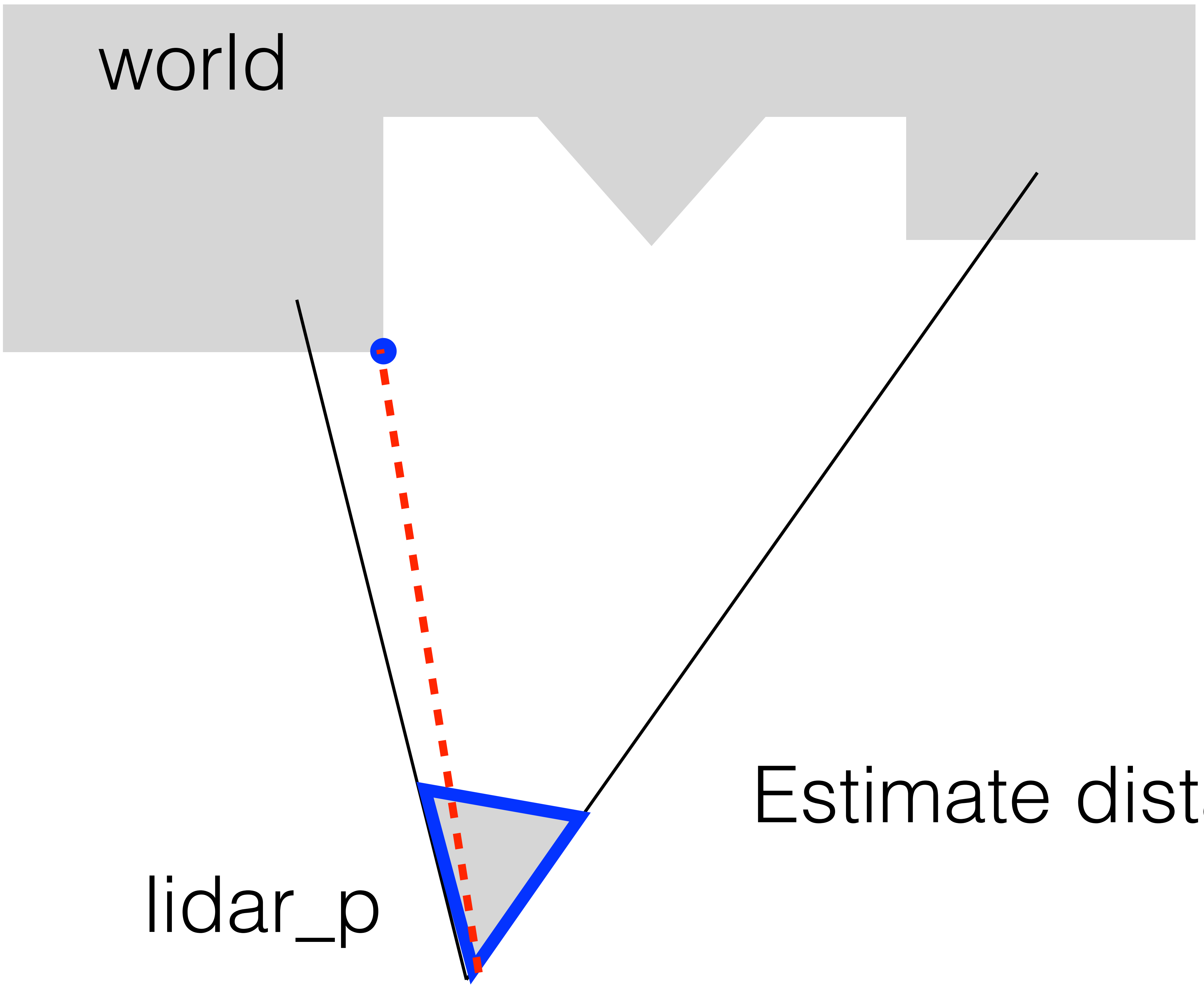
The topic of this lecture:
- estimating transformation between **static** coord. frames (sensor calibration)
- principle of lidar, camera, realsense, stereo …

The topic of next lecture:
- estimating transformation between **dynamic** coord. frames (robot/sensor localization - SLAM)

Lidar



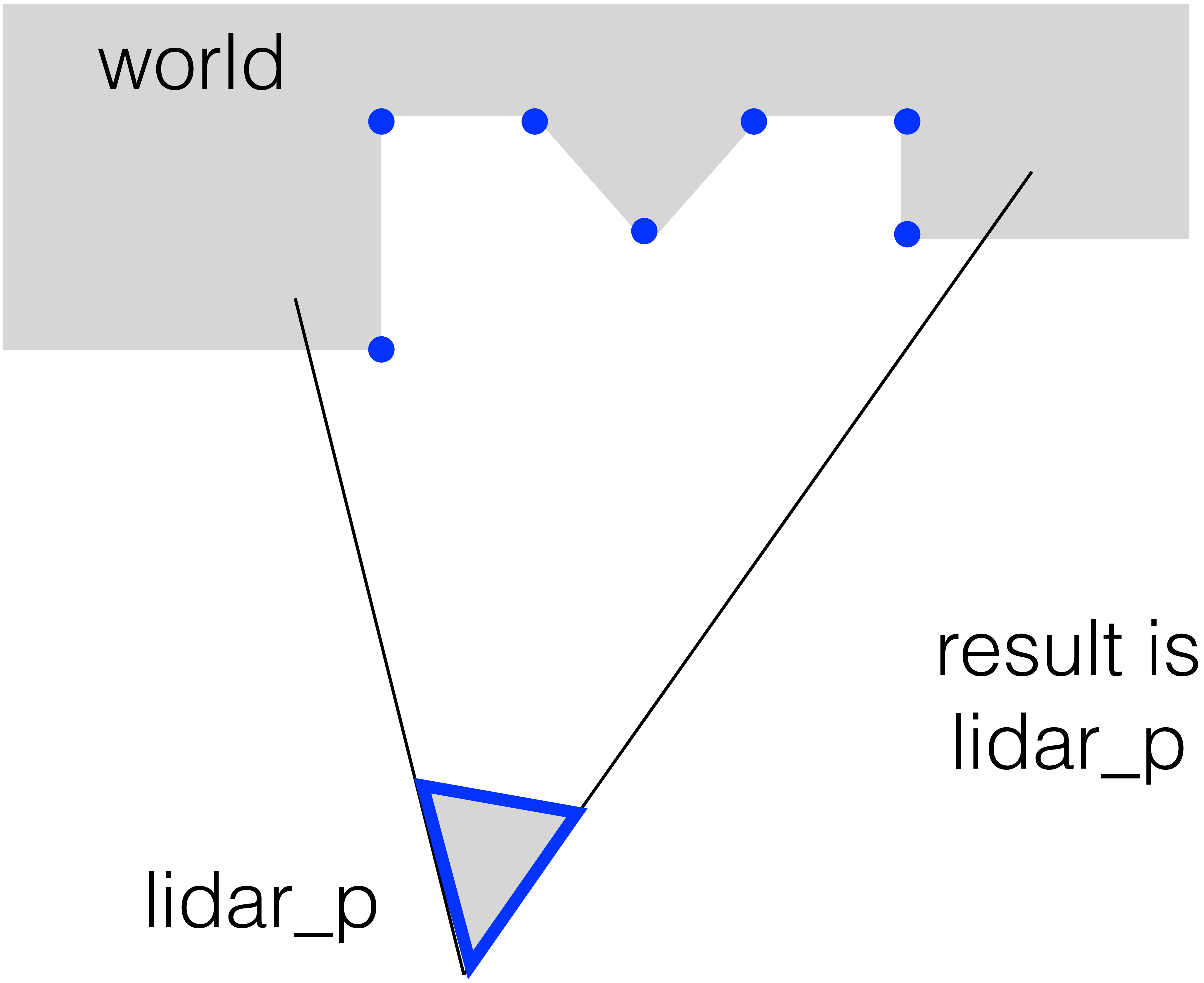
Lidar



Estimate distance from time-of-flight.

$$d = c \cdot \frac{t}{2}$$

Lidar

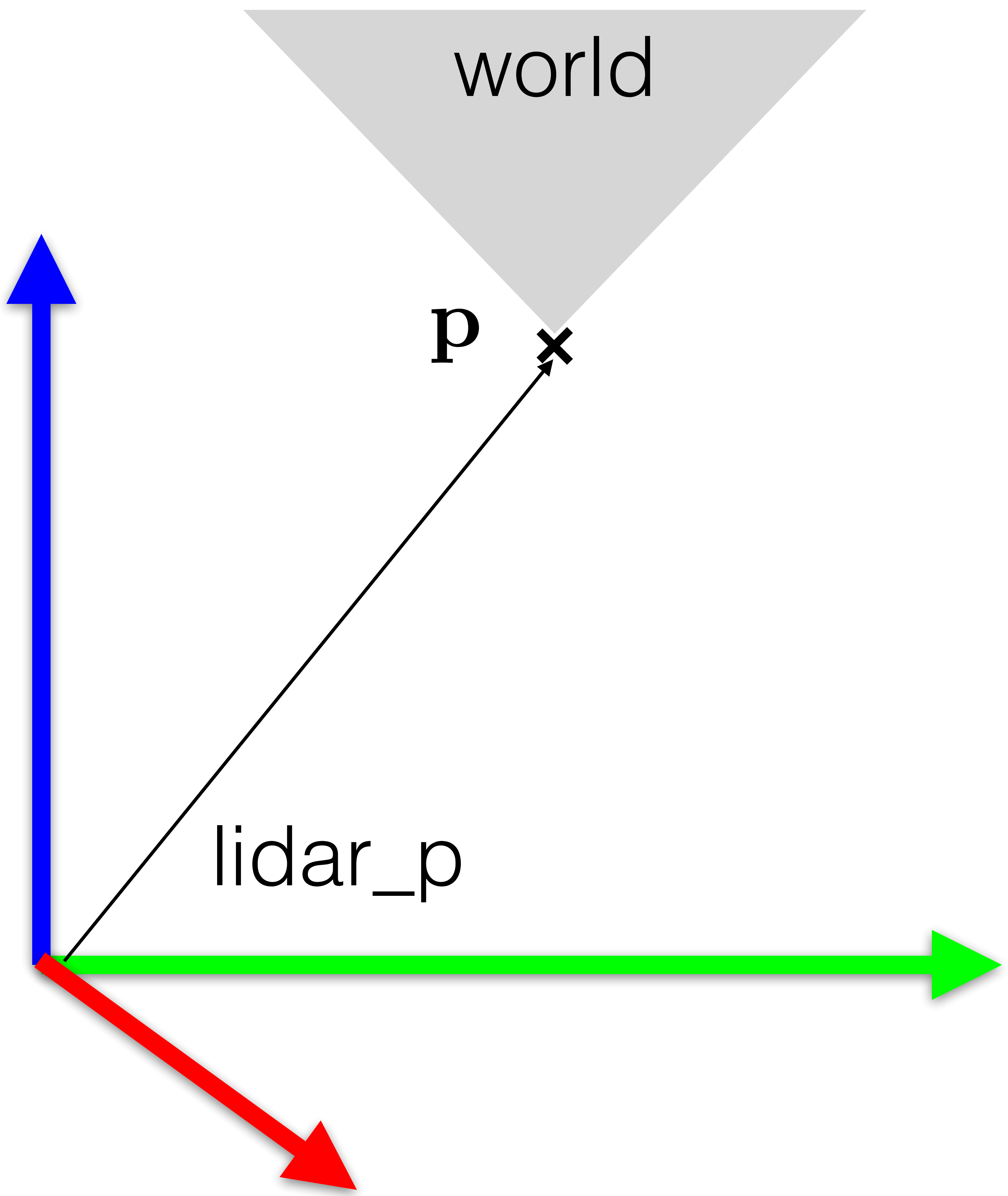


result is 3D point cloud in lidar_p coordinate frame

Euclidean transformation of a rigid body

Let us now work only with two coordinate frames:

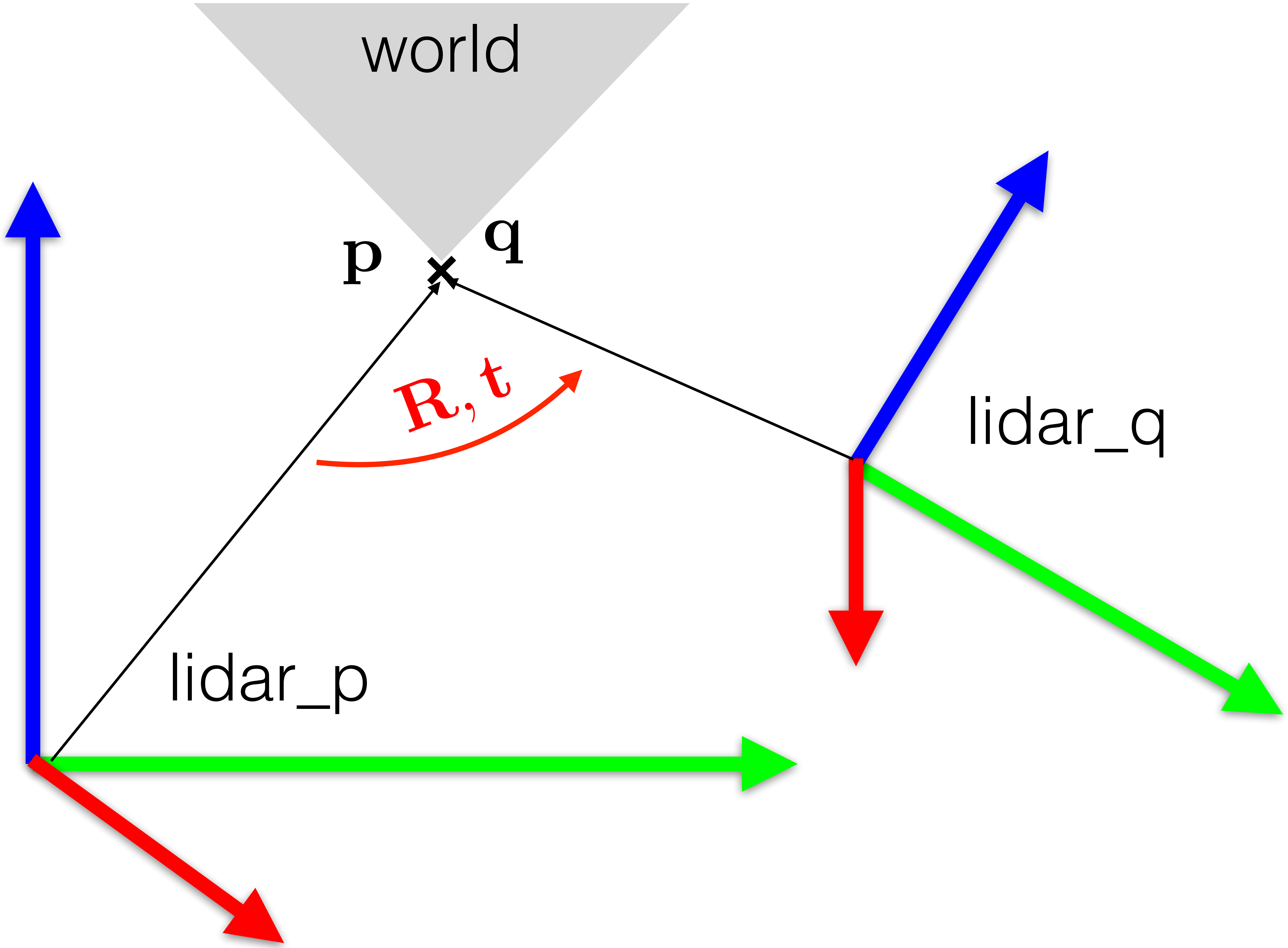
- lidar_p in which points are denoted as $\mathbf{p} \in \mathcal{R}^3$



Euclidean transformation of a rigid body

Let us now work only with two coordinate frames:

- lidar_p in which points are denoted as $\mathbf{p} \in \mathcal{R}^3$
- lidar_q in which points are denoted as $\mathbf{q} \in \mathcal{R}^3$



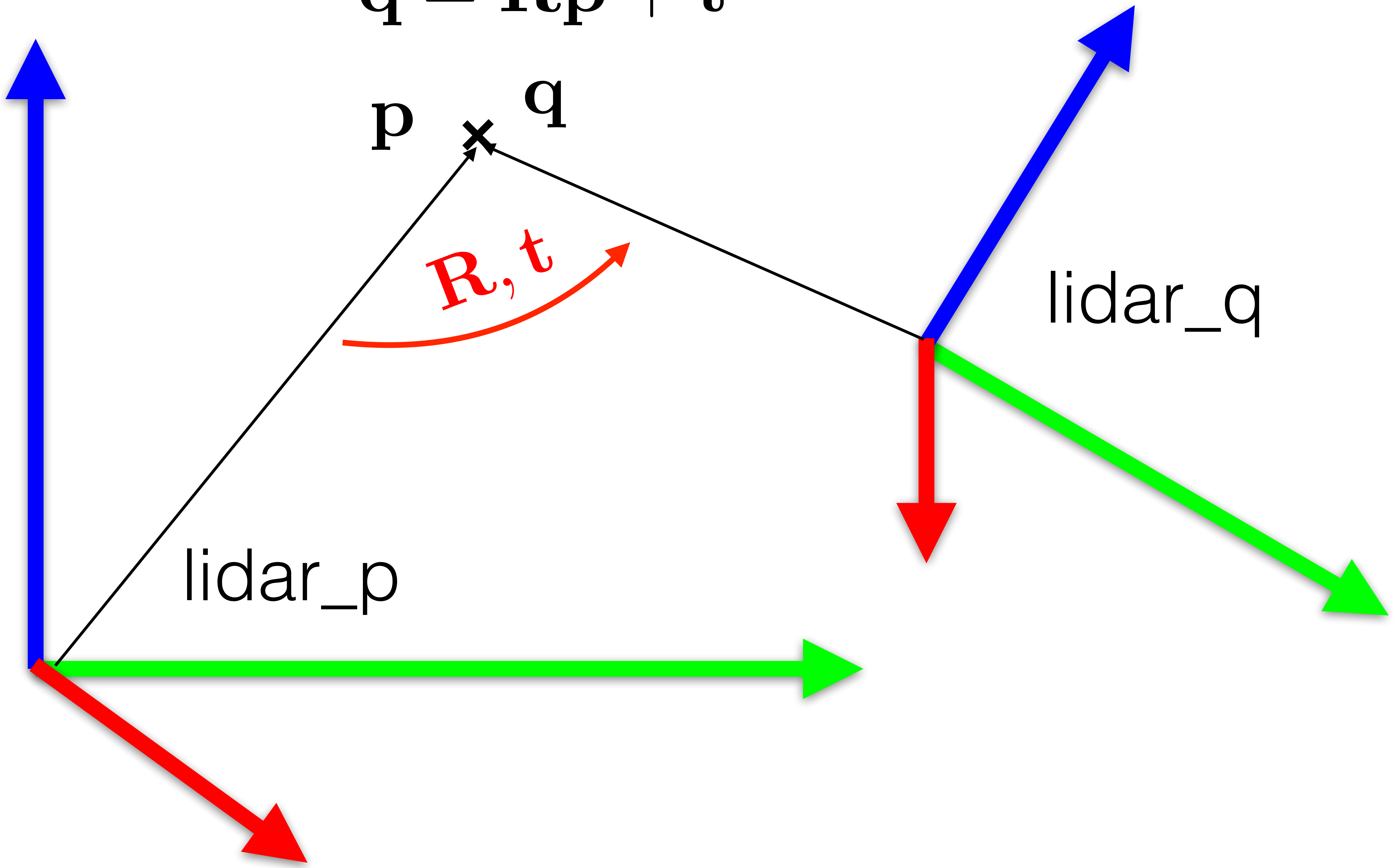
Euclidean transformation of a rigid body

Let us now work only with two coordinate frames:

- lidar_p in which points are denoted as $\mathbf{p} \in \mathcal{R}^3$
- lidar_q in which points are denoted as $\mathbf{q} \in \mathcal{R}^3$

Transformation between measurements uniquely determined:

$$\mathbf{q} = \mathbf{R}\mathbf{p} + \mathbf{t}$$



Euclidean transformation of a rigid body

- Euclidean transformation $\mathbf{q} = \mathbf{R}\mathbf{p} + \mathbf{t}$

  where $\mathbf{t} \in \mathcal{R}^3 \quad \mathbf{R} \in \mathcal{SO}(3)$

  $\mathcal{SO}(3)_= \{\mathbf{R} \in \mathcal{R}^{3\times3} \mid \mathbf{R}^\top \mathbf{R} = \mathbf{I},\ \det(\mathbf{R}) = +1\}$

Euclidean transformation of a rigid body in **homogeneous coordinates**

- The representation $\mathbf{p}$ of a geometric object is homogeneous iff $\mathbf{p}$ and $\lambda\mathbf{p}$ represent the same object for any $\lambda \neq 0$.

Euclidean → Homogeneous (projective)

$$\mathbf{p} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \quad \rightarrow \quad \overline{\mathbf{p}} = \begin{bmatrix} \mathbf{p} \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

$$\mathbf{p} = \begin{bmatrix} a/d \\ b/d \\ c/d \end{bmatrix} \quad \leftarrow \quad \overline{\mathbf{p}} = \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix}$$

- Euclidean transformation is given by matrix $\overline{\mathbf{q}} = \mathbf{M}\,\overline{\mathbf{p}}$

where $\mathbf{M} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}^\top & 1 \end{bmatrix} \quad \overline{\mathbf{p}} = \begin{bmatrix} \mathbf{p} \\ 1 \end{bmatrix} \quad \overline{\mathbf{q}} = \begin{bmatrix} \mathbf{q} \\ 1 \end{bmatrix}$

https://en.wikipedia.org/wiki/August_Ferdinand_M%C3%B6bius

Euclidean transformation of a rigid body in **homogeneous coordinates**

Example 1:

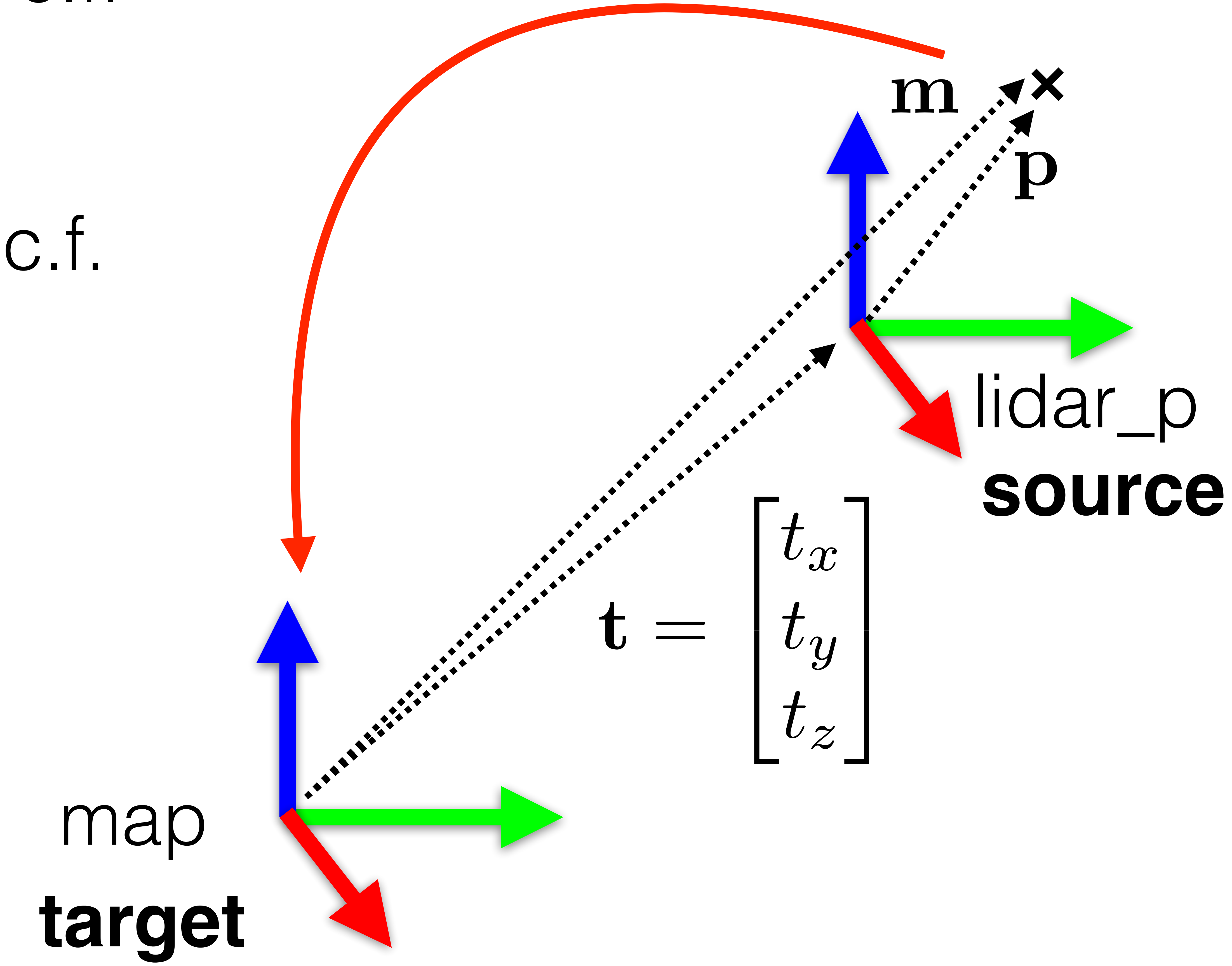
What is the meaning of $\mathbf{M}_{mp}$???

$$\mathbf{M}_{mp} = \begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

1. $\mathbf{M}_{mp}$ transfers $\mathbf{p}$ from source c.f. to target c.f.

   $\overline{\mathbf{m}} = \mathbf{M}_{mp}\overline{\mathbf{p}}$

2. $\mathbf{M}_{mp}$ contains pose of source c.f. in target c.f.

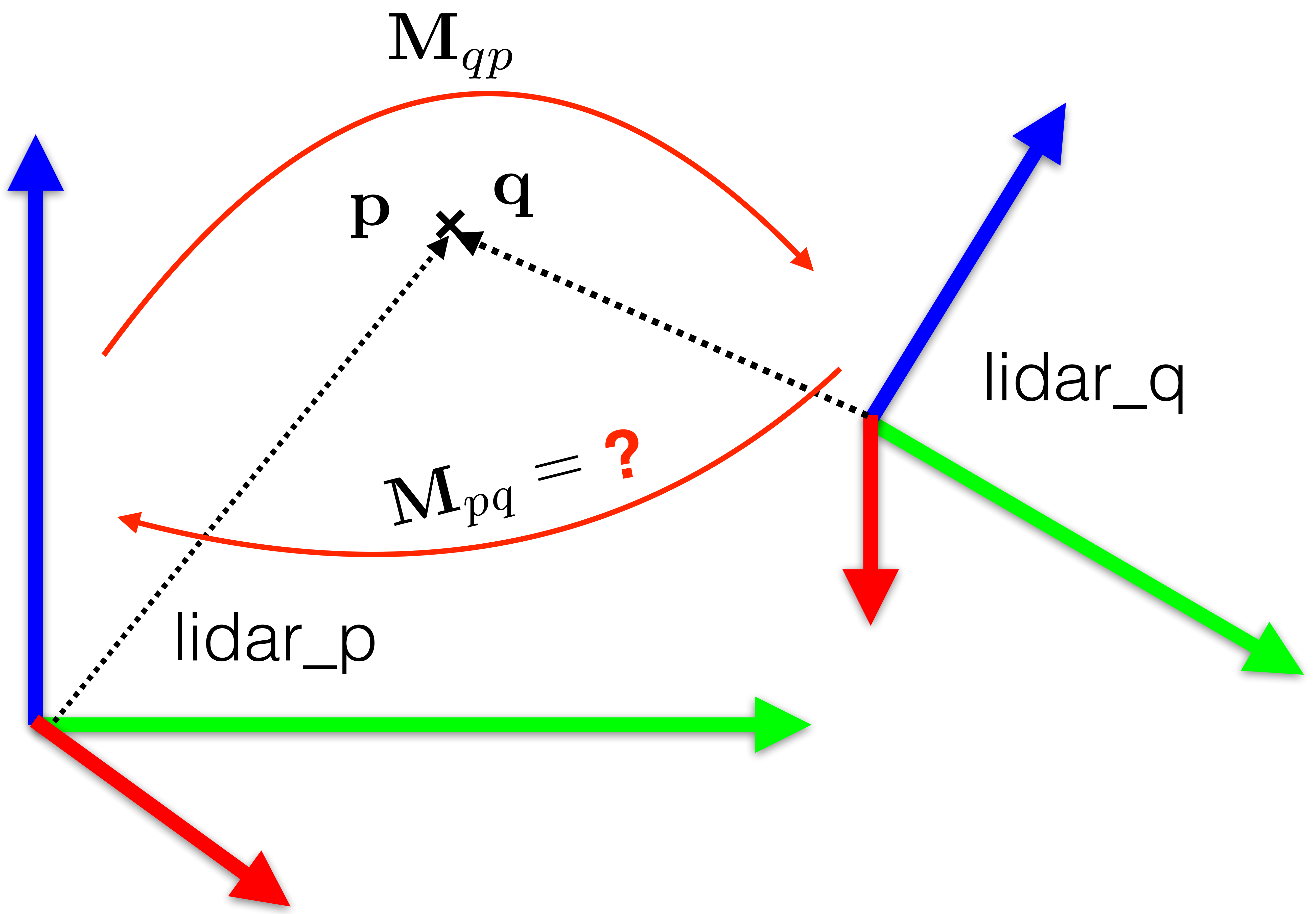


Euclidean transformation of a rigid body in **homogeneous coordinates**

Example 2:

Given transformation from lidar_p to lidar_q $\mathbf{M}_{qp}$
what is inverse transformation $\mathbf{M}_{pq}$?

$$\mathbf{M}_{qp} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ 000 & 1 \end{bmatrix}$$



Euclidean transformation of a rigid body in **homogeneous coordinates**

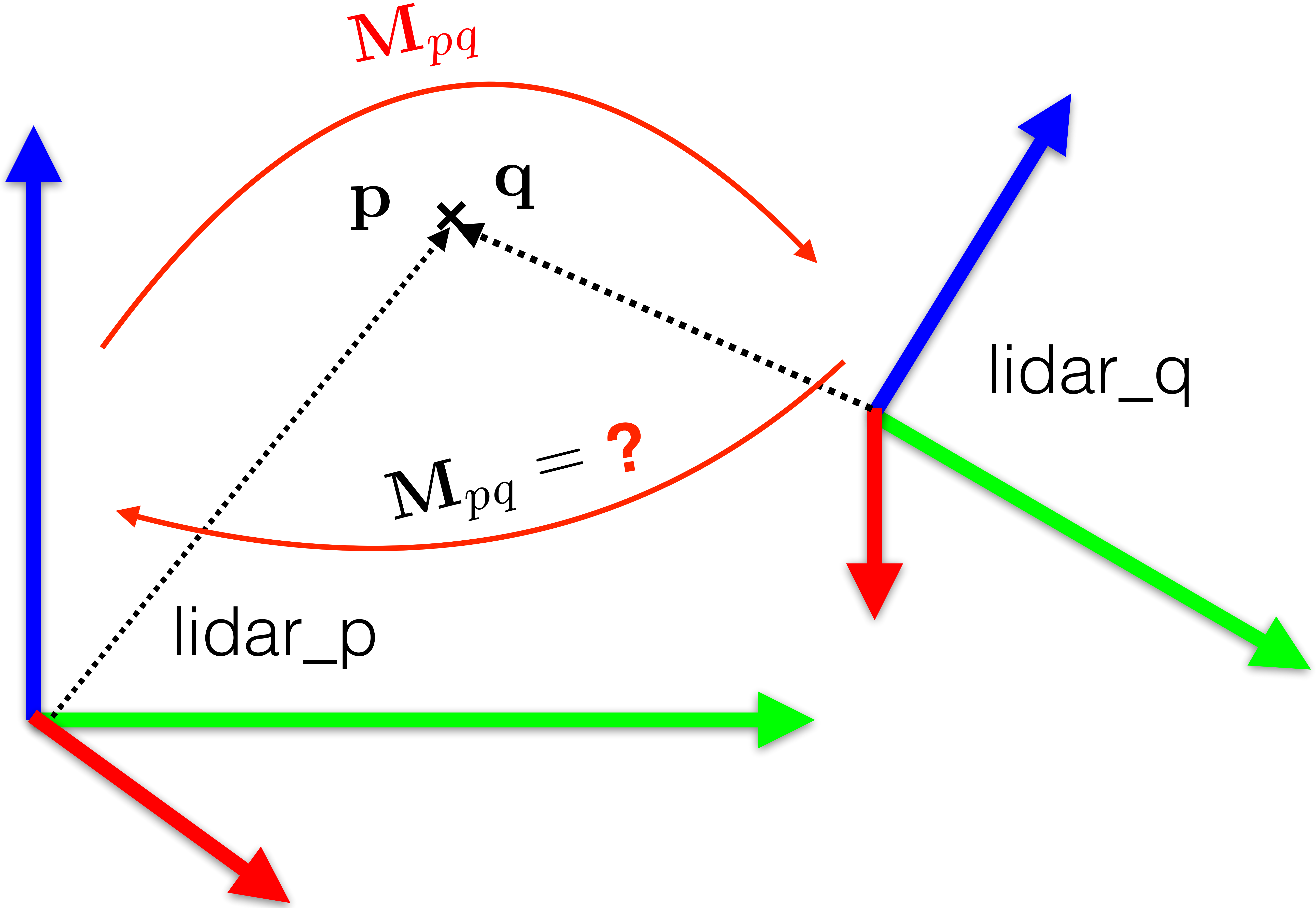
Example 2:

Given transformation from lidar_p to lidar_q $\mathbf{M}_{qp}$,
what is inverse transformation $\mathbf{M}_{pq}$?

$$\mathbf{M}_{qp} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ 000 & 1 \end{bmatrix}$$

$$\mathbf{M}_{pq} = \begin{bmatrix} \mathbf{R}^\top & -\mathbf{R}^\top\mathbf{t} \\ 000 & 1 \end{bmatrix}$$

$$= \mathbf{M}_{qp}^{-1}$$



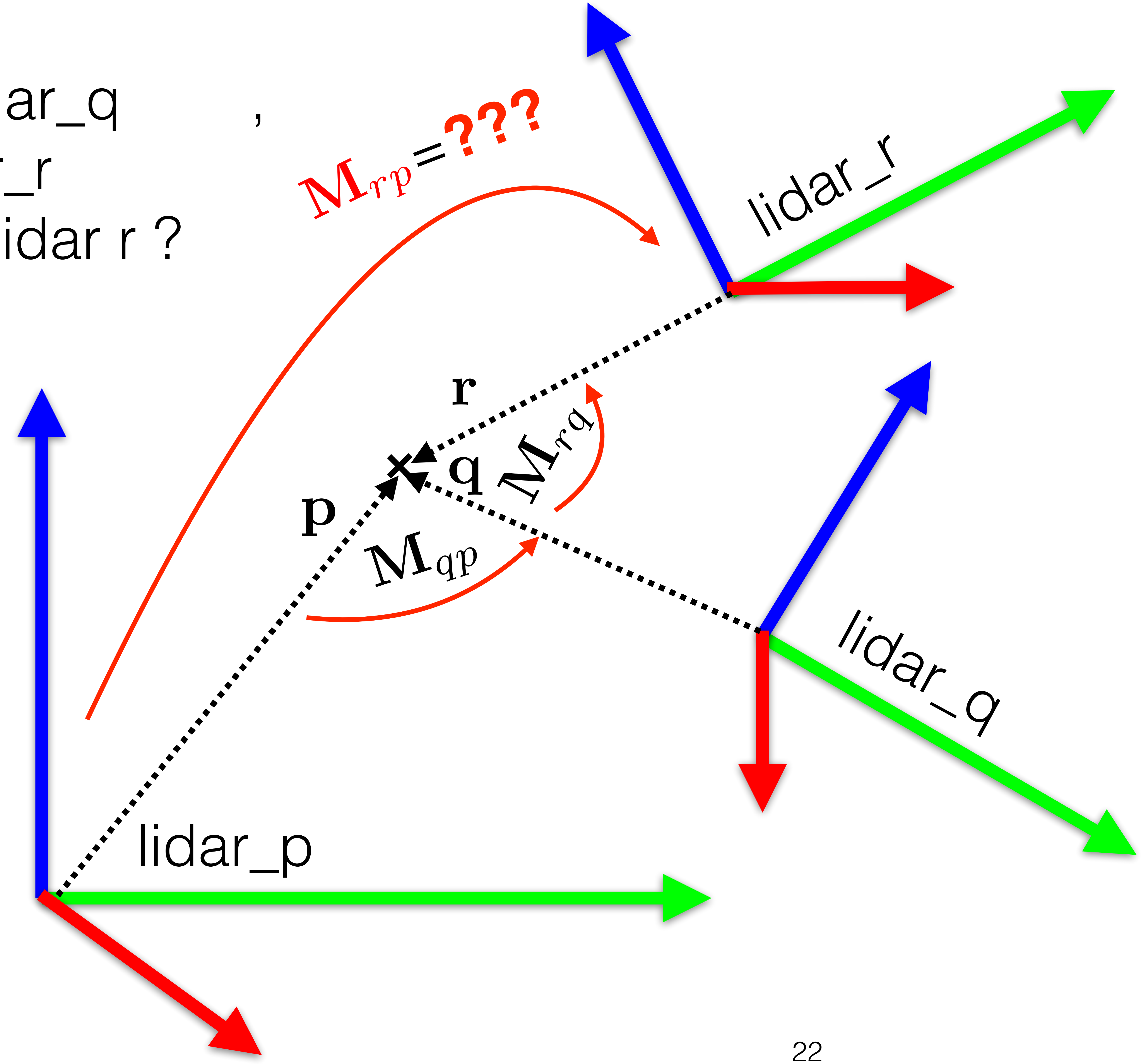
Euclidean transformation of a rigid body in **homogeneous coordinates**

Example 3:

Given transformation from lidar_p to lidar_q ,
and transformation from lidar_q to lidar_r
what is transformation from lidar_p to lidar r ?

$$\mathbf{M}_{rp} = \mathbf{M}_{rq}\mathbf{M}_{qp} =$$

$$= \begin{bmatrix} \mathbf{R}_{rq}\mathbf{R}_{qp} & \mathbf{R}_{rq}\mathbf{t}_{qp} + \mathbf{t}_{rq} \\ \mathbf{0} & 1 \end{bmatrix}$$



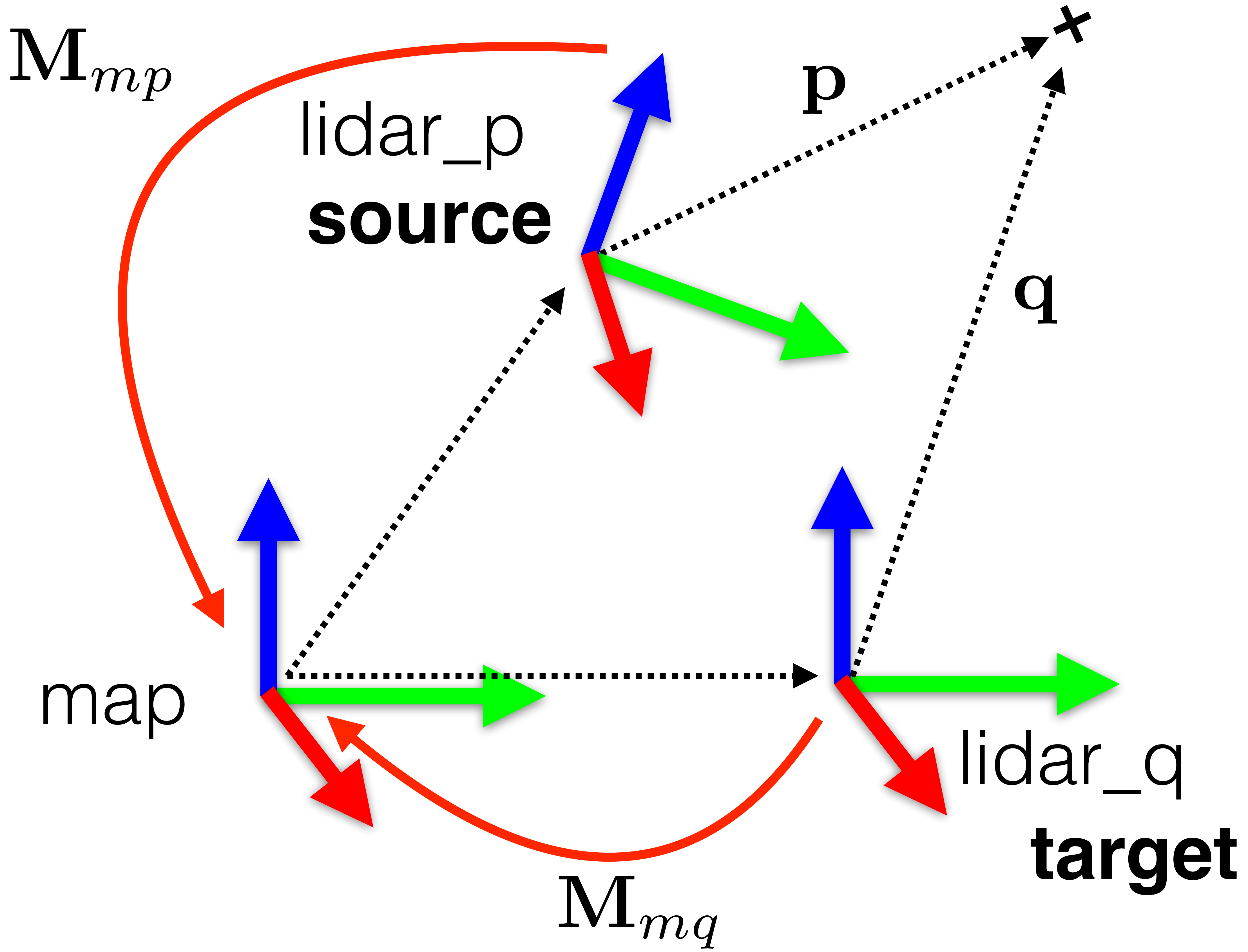
Euclidean transformation of a rigid body in **homogeneous coordinates**

Example 4:

$\mathbf{M}_{qp}$ = **???**

$$\mathbf{M}_{qp} = \mathbf{M}_{qm}\mathbf{M}_{mp} = \mathbf{M}_{mq}^{-1}\mathbf{M}_{mp}$$

What is the meaning of $\mathbf{M}_{qp}$???



Summary: Euclidean transformation of a rigid body

- Euclidean transformation $\mathbf{q} = \mathbf{R}\mathbf{p} + \mathbf{t}$

  where $\mathbf{t} \in \mathcal{R}^3$ $\mathbf{R} \in \mathcal{SO}(3) = \{\mathbf{R} \in \mathcal{R}^{3\times 3} \mid \mathbf{R}^\top\mathbf{R} = \mathbf{I},\ \det(\mathbf{R}) = +1\}$

- Euclidean transformation in h.c. $\overline{\mathbf{q}} = \mathbf{M}\,\overline{\mathbf{p}}$ is given by 4x4 matrix

  where $\mathbf{M} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}^\top & 1 \end{bmatrix}$ $\overline{\mathbf{p}} = \begin{bmatrix} \mathbf{p} \\ 1 \end{bmatrix}$ $\overline{\mathbf{q}} = \begin{bmatrix} \mathbf{q} \\ 1 \end{bmatrix}$

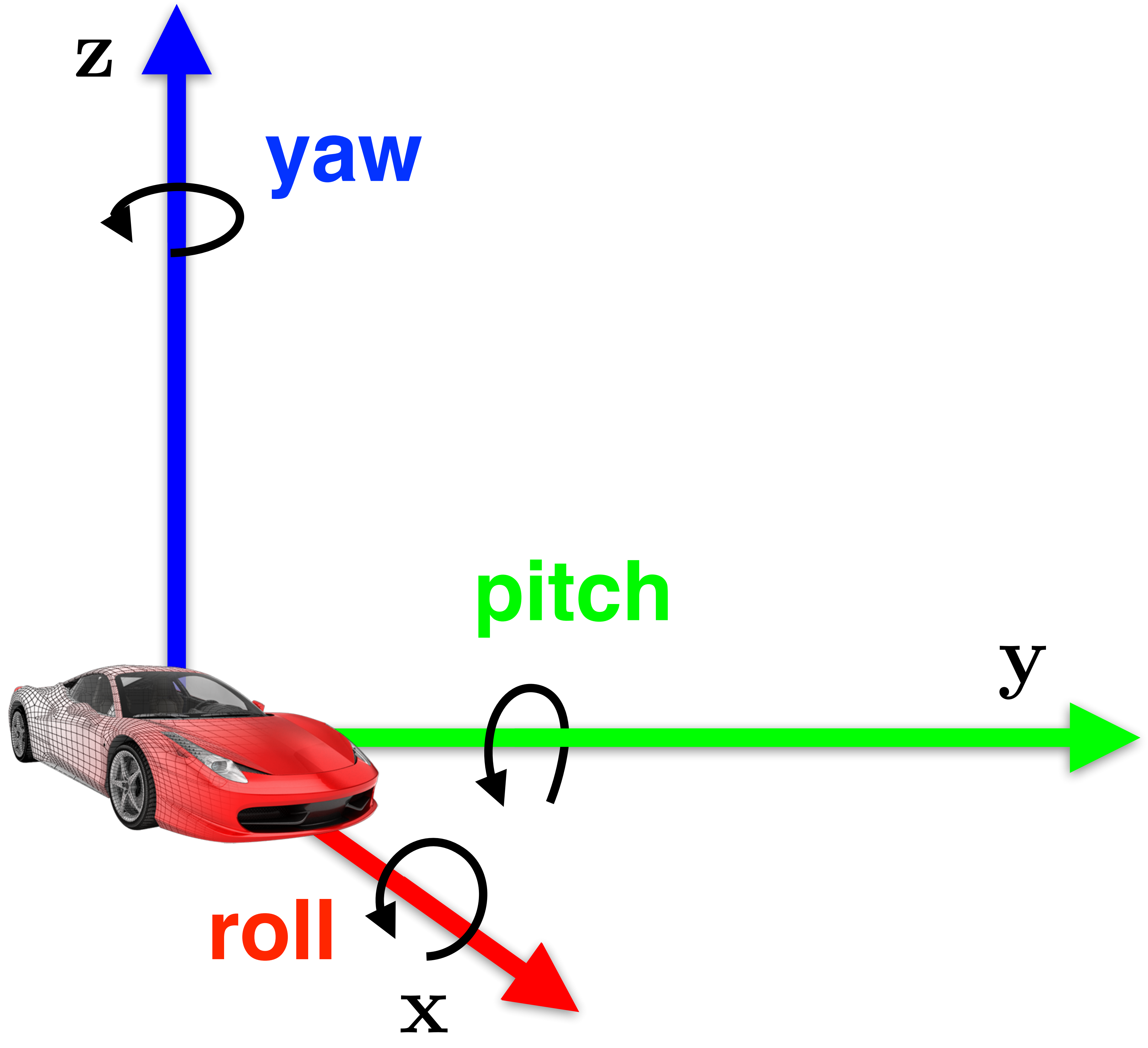
- Set of all transformations forms Special Euclidean group under matrix multiplic,

$$\mathcal{SE}(3) = \{\mathbf{M} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix} \mid \mathbf{R} \in \mathcal{SO}(3), \mathbf{t} \in \mathcal{R}^3\}$$

- Since SO(3) constraint is sometimes unsuitable (3-dim manifold in 9-dim space)
- Alternative representations are also used.

Different representations of the rotation: Euler angles

Any rotation can be achieved by 3 successive rotations around axes of coord. s.



1. roll: $$\mathbf{R}_x(\gamma) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\gamma & -\sin\gamma \\ 0 & \sin\gamma & \cos\gamma \end{bmatrix}$$

2. pitch: $$\mathbf{R}_y(\beta) = \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix}$$

3. yaw: $$\mathbf{R}_z(\alpha) = \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{R}(\alpha, \beta, \gamma) = \mathbf{R}_z(\alpha)\mathbf{R}_y(\beta)\mathbf{R}_x(\gamma)$$

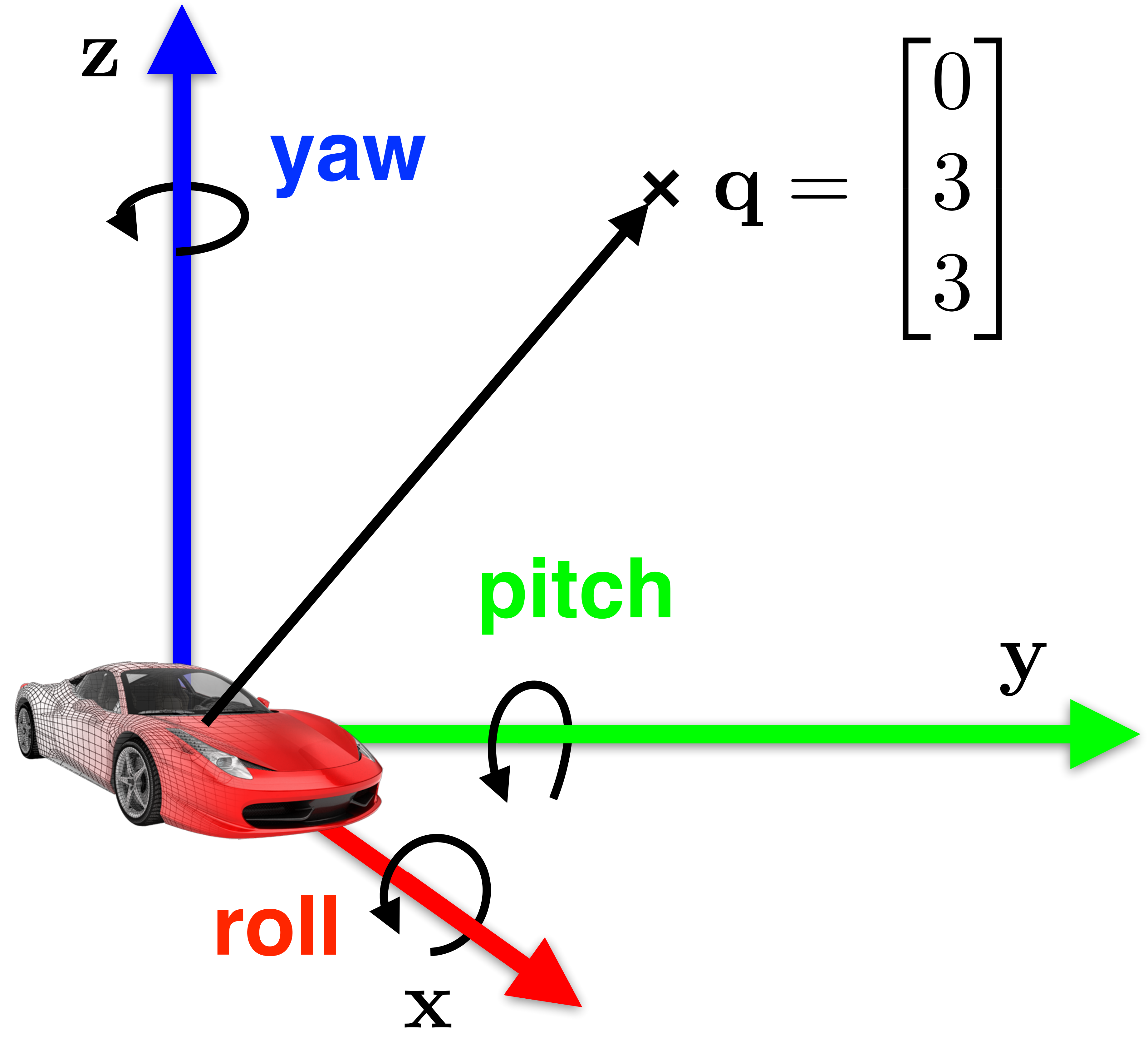
Different representations of the rotation: Euler angles

Any rotation can be achieved by 3 successive rotations around axes of coord. s.

Example 1:

What is 90-rotation around axis y?

What does this rotation preserve?



$$\mathbf{q} = \begin{bmatrix} 0 \\ 3 \\ 3 \end{bmatrix}$$

1. roll: $$\mathbf{R}_x(\gamma) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\gamma & -\sin\gamma \\ 0 & \sin\gamma & \cos\gamma \end{bmatrix}$$

2. pitch: $$\mathbf{R}_y(\beta) = \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix}$$

3. yaw: $$\mathbf{R}_z(\alpha) = \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{R}(\alpha, \beta, \gamma) = \mathbf{R}_z(\alpha)\mathbf{R}_y(\beta)\mathbf{R}_x(\gamma)$$

Different representations of the rotation: Euler angles

Any rotation can be achieved by 3 successive rotations around axes of coord. s.

Example 1:

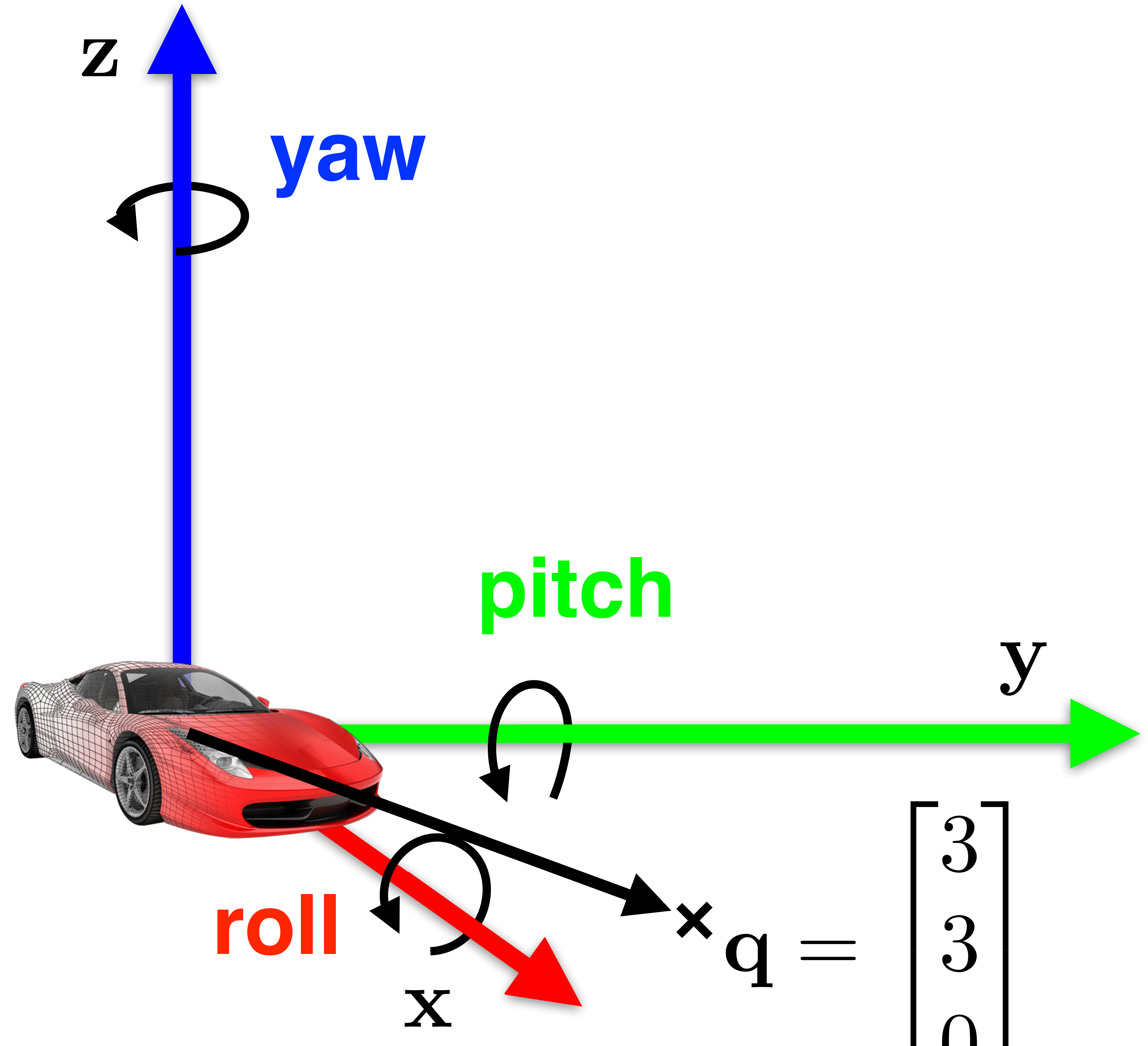
What is 90-rotation around axis y?

What does this rotation preserve?

1. roll: $\mathbf{R}_x(\gamma) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\gamma & -\sin\gamma \\ 0 & \sin\gamma & \cos\gamma \end{bmatrix}$

2. pitch: $\mathbf{R}_y(\beta) = \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix}$

3. yaw: $\mathbf{R}_z(\alpha) = \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix}$



$\mathbf{q} = \begin{bmatrix} 3 \\ 3 \\ 0 \end{bmatrix}$

$$\mathbf{R}(\alpha, \beta, \gamma) = \mathbf{R}_z(\alpha)\mathbf{R}_y(\beta)\mathbf{R}_x(\gamma)$$

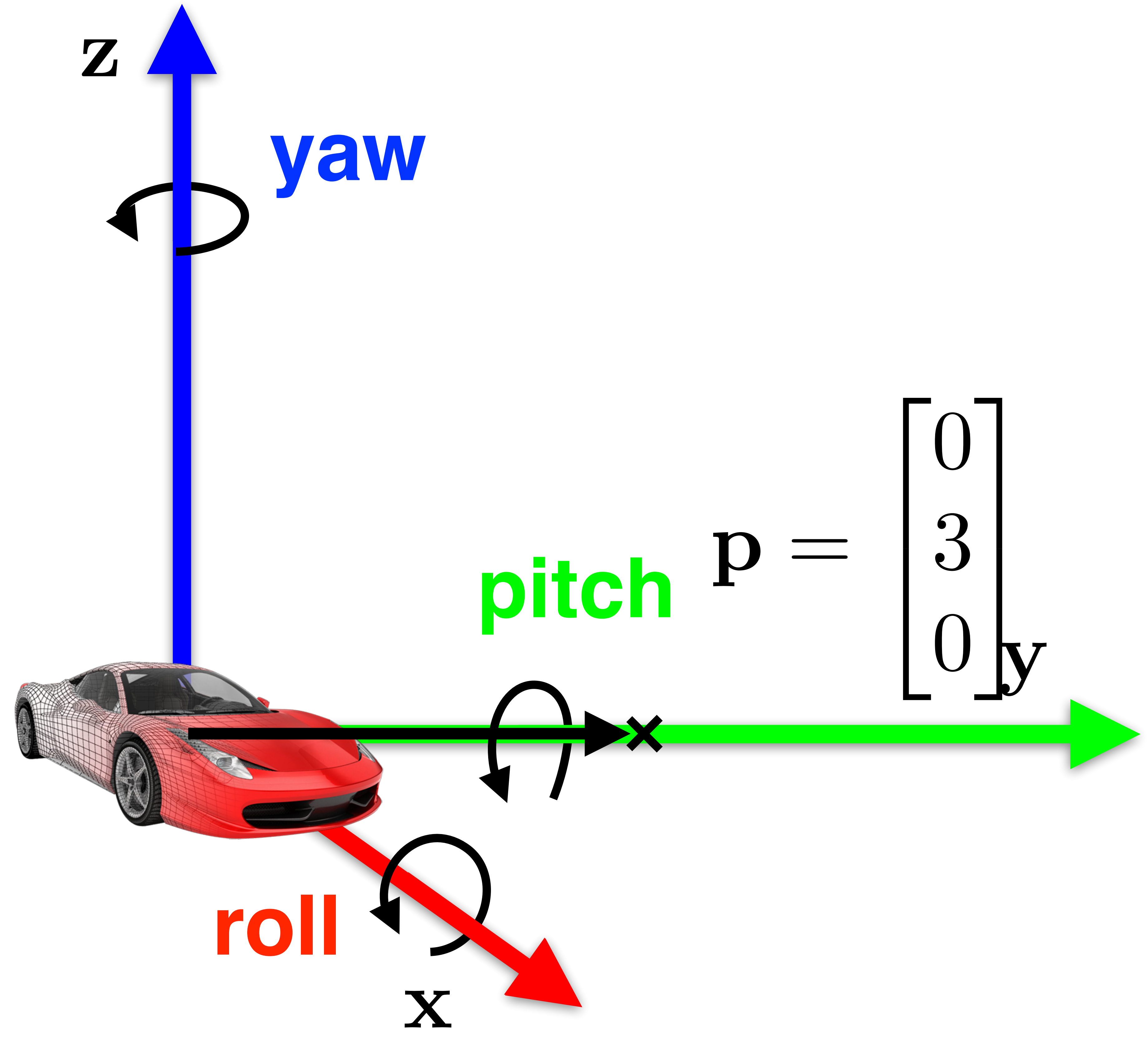
Different representations of the rotation: Euler angles

Any rotation can be achieved by 3 successive rotations around axes of coord. s.

Example 1:

What is 90-rotation around axis y?

What does this rotation preserve?



$$\mathbf{p} = \begin{bmatrix} 0 \\ 3 \\ 0 \end{bmatrix}$$

1. roll: $$\mathbf{R}_x(\gamma) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\gamma & -\sin\gamma \\ 0 & \sin\gamma & \cos\gamma \end{bmatrix}$$

2. pitch: $$\mathbf{R}_y(\beta) = \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix}$$

3. yaw: $$\mathbf{R}_z(\alpha) = \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{R}(\alpha, \beta, \gamma) = \mathbf{R}_z(\alpha)\mathbf{R}_y(\beta)\mathbf{R}_x(\gamma)$$

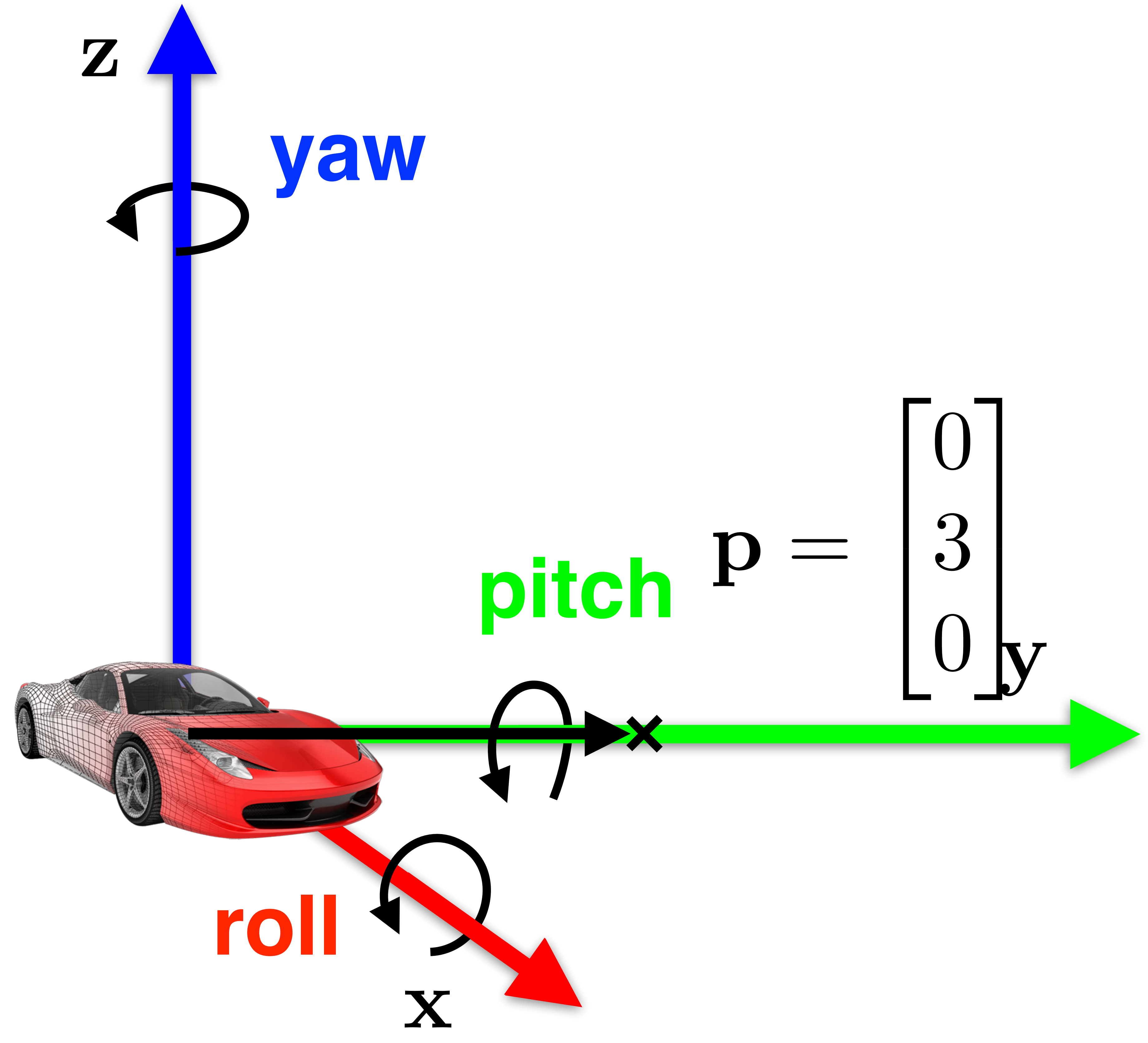
Different representations of the rotation: Euler angles

Any rotation can be achieved by 3 successive rotations around axes of coord. s.

Example 1:

What is 90-rotation around axis y?

What does this rotation preserve?



$$\mathbf{p} = \begin{bmatrix} 0 \\ 3 \\ 0 \end{bmatrix}$$

1. roll: $\mathbf{R}_x(\gamma) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\gamma & -\sin\gamma \\ 0 & \sin\gamma & \cos\gamma \end{bmatrix}$

2. pitch: $\mathbf{R}_y(\beta) = \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix}$

3. yaw: $\mathbf{R}_z(\alpha) = \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix}$

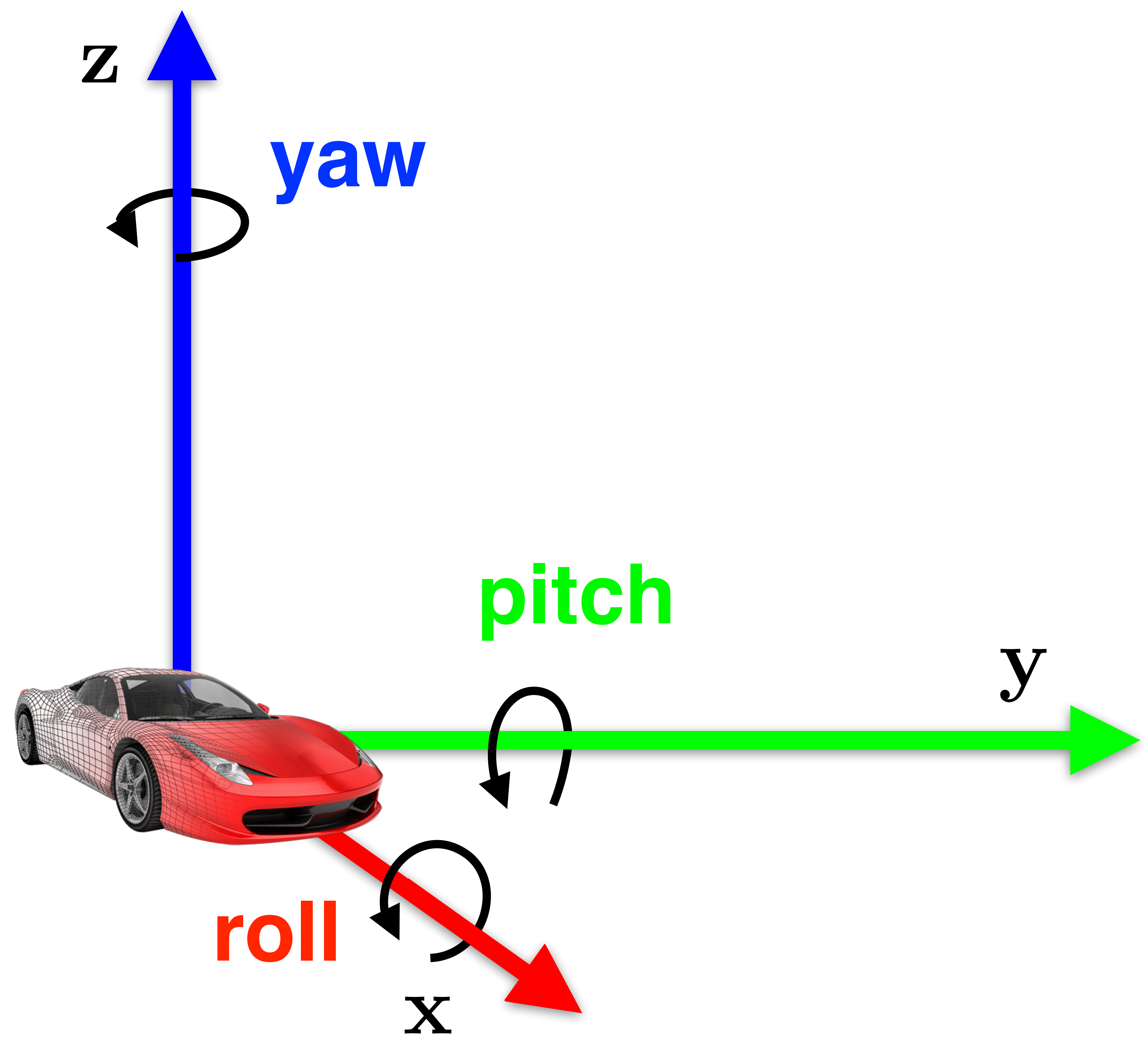
$$\mathbf{R}(\alpha, \beta, \gamma) = \mathbf{R}_z(\alpha)\mathbf{R}_y(\beta)\mathbf{R}_x(\gamma)$$

Different representations of the rotation: Euler angles

Any rotation can be achieved by 3 successive rotations around axes of coord. s.

roll pitch yaw

Gimbal lock: [**:**, **90**, **0**]
[**0**, **90**, **:**]



1. roll: $\mathbf{R}_x(\gamma) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\gamma & -\sin\gamma \\ 0 & \sin\gamma & \cos\gamma \end{bmatrix}$

2. pitch: $\mathbf{R}_y(\beta) = \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix}$

3. yaw: $\mathbf{R}_z(\alpha) = \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix}$

$$\mathbf{R}(\alpha, \beta, \gamma) = \mathbf{R}_z(\alpha)\mathbf{R}_y(\beta)\mathbf{R}_x(\gamma)$$

Different representations of the rotation: Euler angles

Any rotation can be achieved by 3 successive rotations around axes of coord. s.

roll pitch yaw

Gimbal lock: [**:**, **90**, **0**]
[**0**, **90**, **:**]

1. roll: **2. pitch:** **3. yaw:**

$$\mathbf{R}(\alpha,\beta,\gamma) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\gamma & -\sin\gamma \\ 0 & \sin\gamma & \cos\gamma \end{bmatrix} \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix} \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Different representations of the rotation: Euler angles

Any rotation can be achieved by 3 successive rotations around axes of coord. s.

roll pitch yaw

Gimbal lock: [**:**, **90**, **0**]
[**0**, **90**, **:**]

1. roll: **2. pitch:** **3. yaw:**

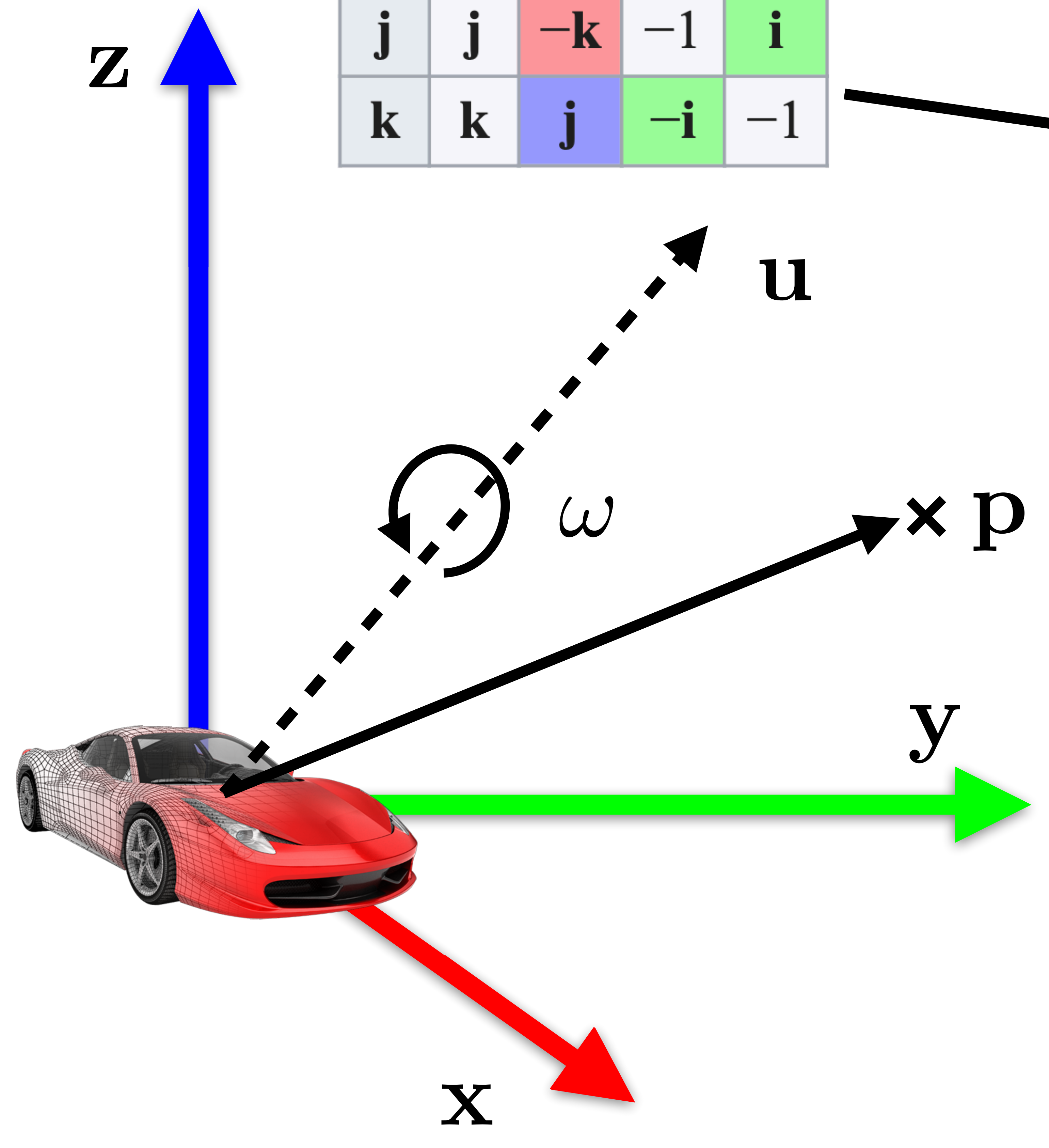
$$\mathbf{R}(\alpha,\beta,\gamma) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\gamma & -\sin\gamma \\ 0 & \sin\gamma & \cos\gamma \end{bmatrix} \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ -1 & 0 & 0 \end{bmatrix} \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 0 & 1 \\ \sin(\alpha+\gamma) & \cos(\alpha+\gamma) & 0 \\ -\cos(\alpha+\gamma) & \sin(\alpha+\gamma) & 0 \end{bmatrix}$$

Different representations of the rotation: Quaternions

Euler rotation theorem: Any sequence of rotations is equivalent to single rotation around fixed axis.

×	1	i	j	k
1	1	i	j	k
i	i	−1	k	−j
j	j	−k	−1	i
k	k	j	−i	−1



$\mathbf{p} \in \mathbb{R}^3$ $\quad \mathbf{p} = (p_x, p_y, p_z) = p_x\mathbf{i} + p_y\mathbf{j} + p_z\mathbf{k}$

$\mathbf{i}, \mathbf{j}, \mathbf{k}$ … cartesian axes

$\mathbb{p} \in \mathbb{H}$ $\quad \mathbb{p} = (0, p_x, p_y, p_z) = 0 + p_x\hat{\mathbf{i}} + p_y\hat{\mathbf{j}} + p_z\hat{\mathbf{k}}$

$\hat{\mathbf{i}}, \hat{\mathbf{j}}, \hat{\mathbf{k}}$ … fundamental quaternions units such that $\hat{\mathbf{i}}^2 = \hat{\mathbf{j}}^2 = \hat{\mathbf{k}}^2 = \hat{\mathbf{i}}\hat{\mathbf{j}}\hat{\mathbf{k}} = -1$

$\mathbb{q} \in \mathbb{H}$ $\quad \mathbb{q} = \cos\frac{\omega}{2} + (u_x\hat{\mathbf{i}} + u_y\hat{\mathbf{j}} + u_z\hat{\mathbf{k}})\sin\frac{\omega}{2}$

$\mathbb{q}^{-1} \in \mathbb{H}$ $\quad \mathbb{q}^{-1} = \cos\frac{\omega}{2} - (u_x\hat{\mathbf{i}} + u_y\hat{\mathbf{j}} + u_z\hat{\mathbf{k}})\sin\frac{\omega}{2}$

$\mathbb{p}' \in \mathbb{H}$ $\quad \mathbb{p}' = \mathbb{q}\mathbb{p}\mathbb{q}^{-1}$ … rotated point in quaternion

$\mathbb{p}' = (0, p'_x, p'_y, p'_z) \in \mathbb{H} \quad \Rightarrow \quad \mathbf{p}' = (p'_x, p'_y, p'_z) \in \mathbb{R}^3$

Summary: Rotation parameterizations

(1) Rotation matrix $\mathbf{R} \in \mathcal{SO}(3)$

$\mathbf{p}'(\mathbf{R}) = \mathbf{R}\mathbf{p}$ with constrain of $\mathbf{R} \in \mathcal{SO}(3)$

9-dim representation with constraint on SO3 manifold

$$\mathcal{SO}(3) = \{\mathbf{R} \in \mathcal{R}^{3\times 3} \mid \mathbf{R}^\top \mathbf{R} = \mathbf{I},\ \det(\mathbf{R}) = +1\}$$

(2) Euler angles $(\alpha, \beta, \gamma) \in \mathcal{R}^3$ 3-dim representation

$$\mathbf{p}'(\alpha, \beta, \gamma) = \mathbf{R}(\alpha, \beta, \gamma)\mathbf{p}$$

(3) Quaternions $\mathbb{q} \in \mathbb{H}$ 4-dim representation

$$\mathbb{p}'(\mathbb{q}) = \mathbb{q}\mathbb{p}\mathbb{q}^{-1}$$

```
from scipy.spatial.transform import Rotation as Rot
Rot.from_matrix(R).as_euler('xyz')
Rot.from_quat(q).to_matrix('xyz')
```

Summary: Rotation parameterizations

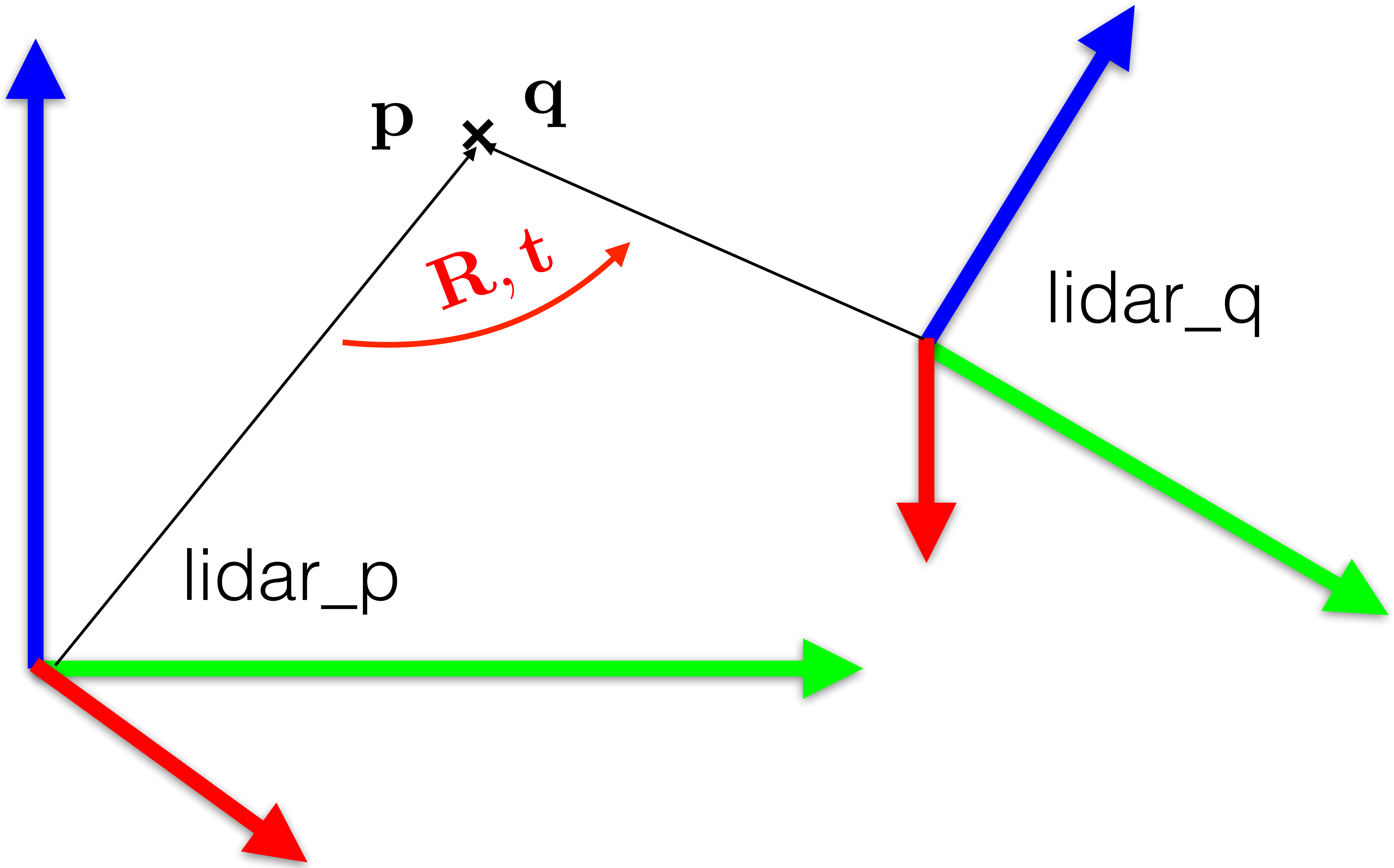
for example: `Rot.from_quat(q).to_matrix('xyz')`

$$\mathbf{R} = \begin{bmatrix} 1 - 2s(q_j^2 + q_k^2) & 2s(q_i q_j - q_k q_r) & 2s(q_i q_k + q_j q_r) \\ 2s(q_i q_j + q_k q_r) & 1 - 2s(q_i^2 + q_k^2) & 2s(q_j q_k - q_i q_r) \\ 2s(q_i q_k - q_j q_r) & 2s(q_j q_k + q_i q_r) & 1 - 2s(q_i^2 + q_j^2) \end{bmatrix}$$

where $s = ||q||^{-2}$

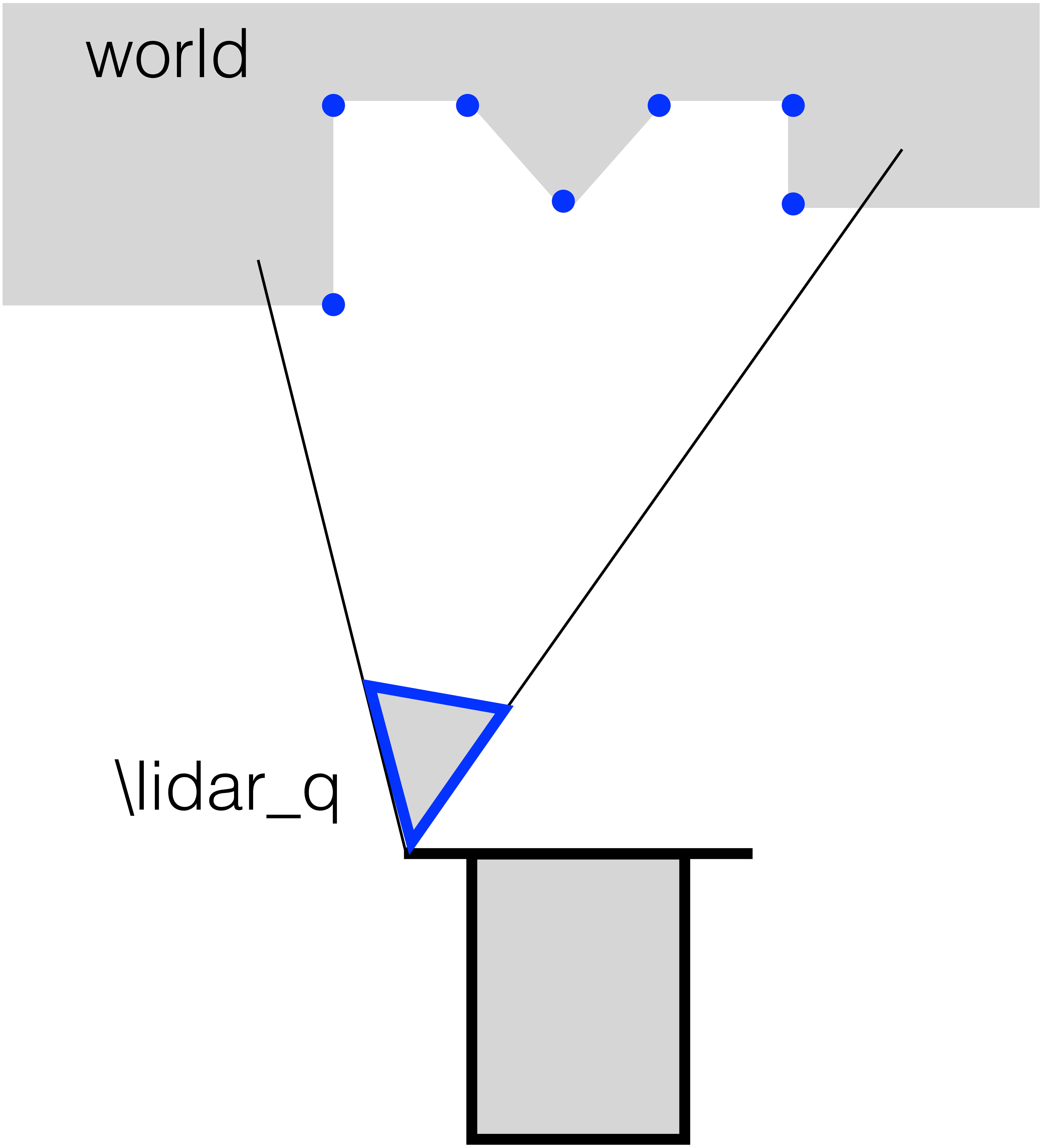
https://en.wikipedia.org/wiki/Quaternions_and_spatial_rotation

Mutual calibration of two coordinate frames

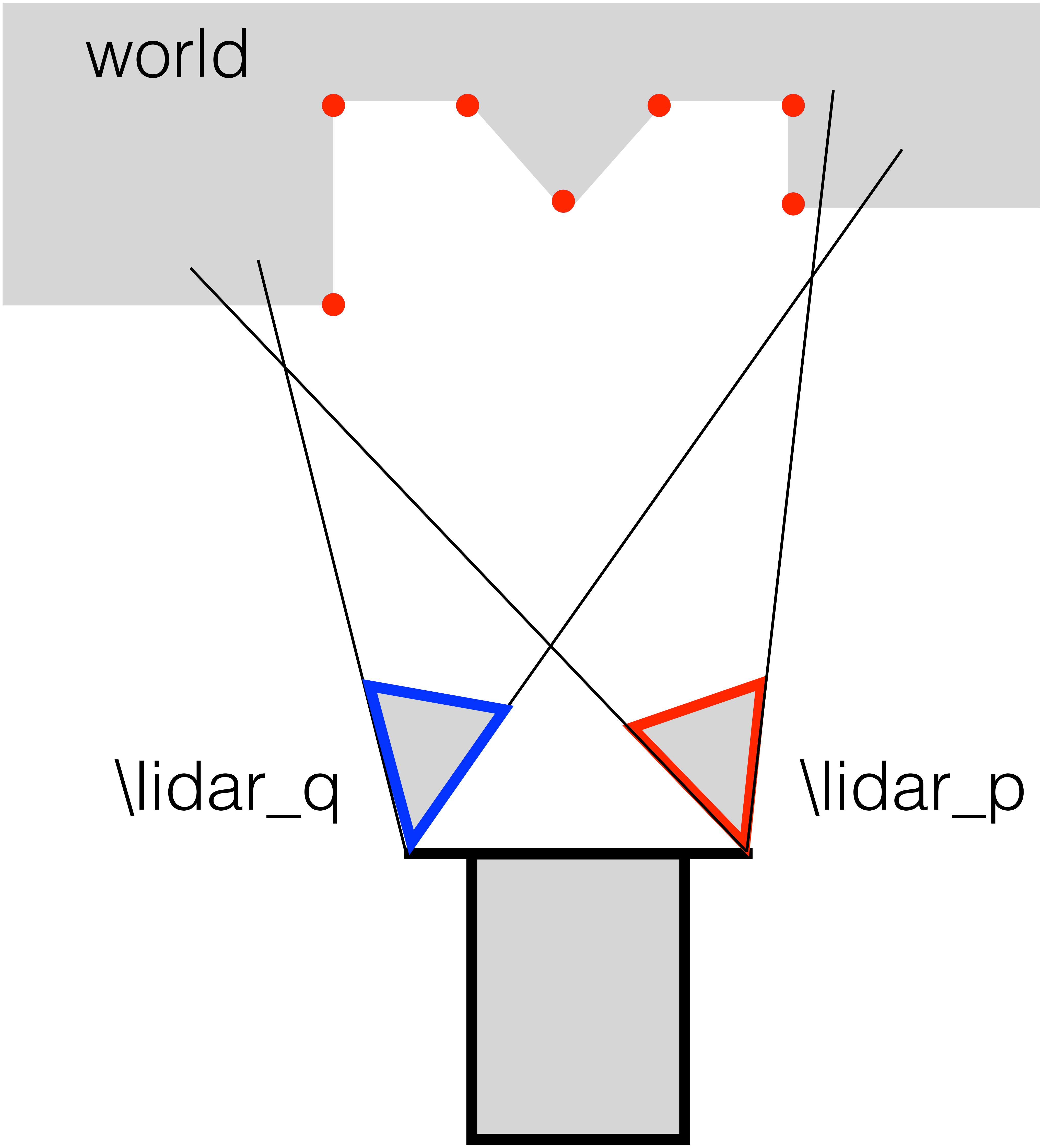


- How do we estimate $\mathbf{R}, \mathbf{t}$?

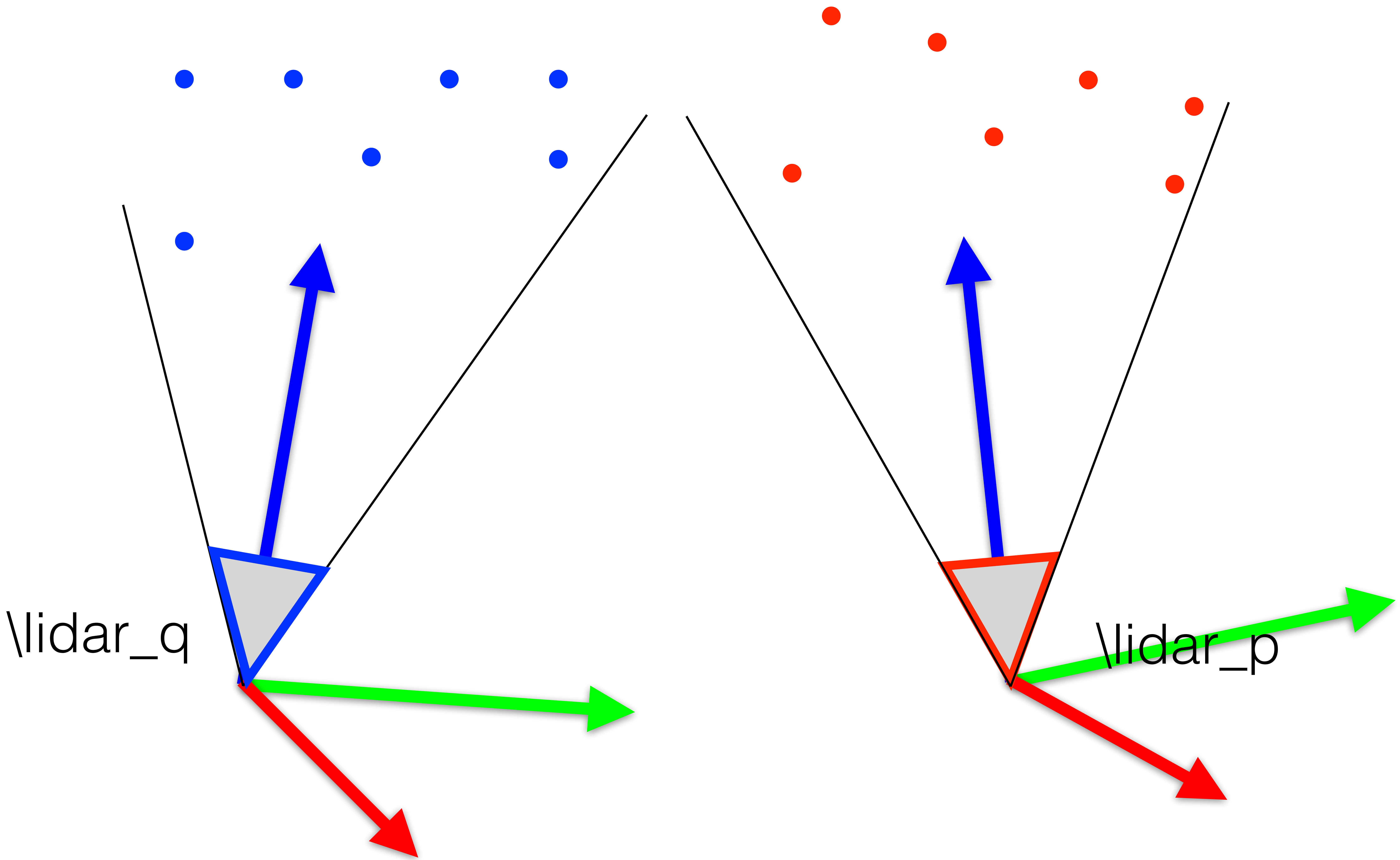
Mutual calibration of two coordinate frames



Mutual calibration of two coordinate frames

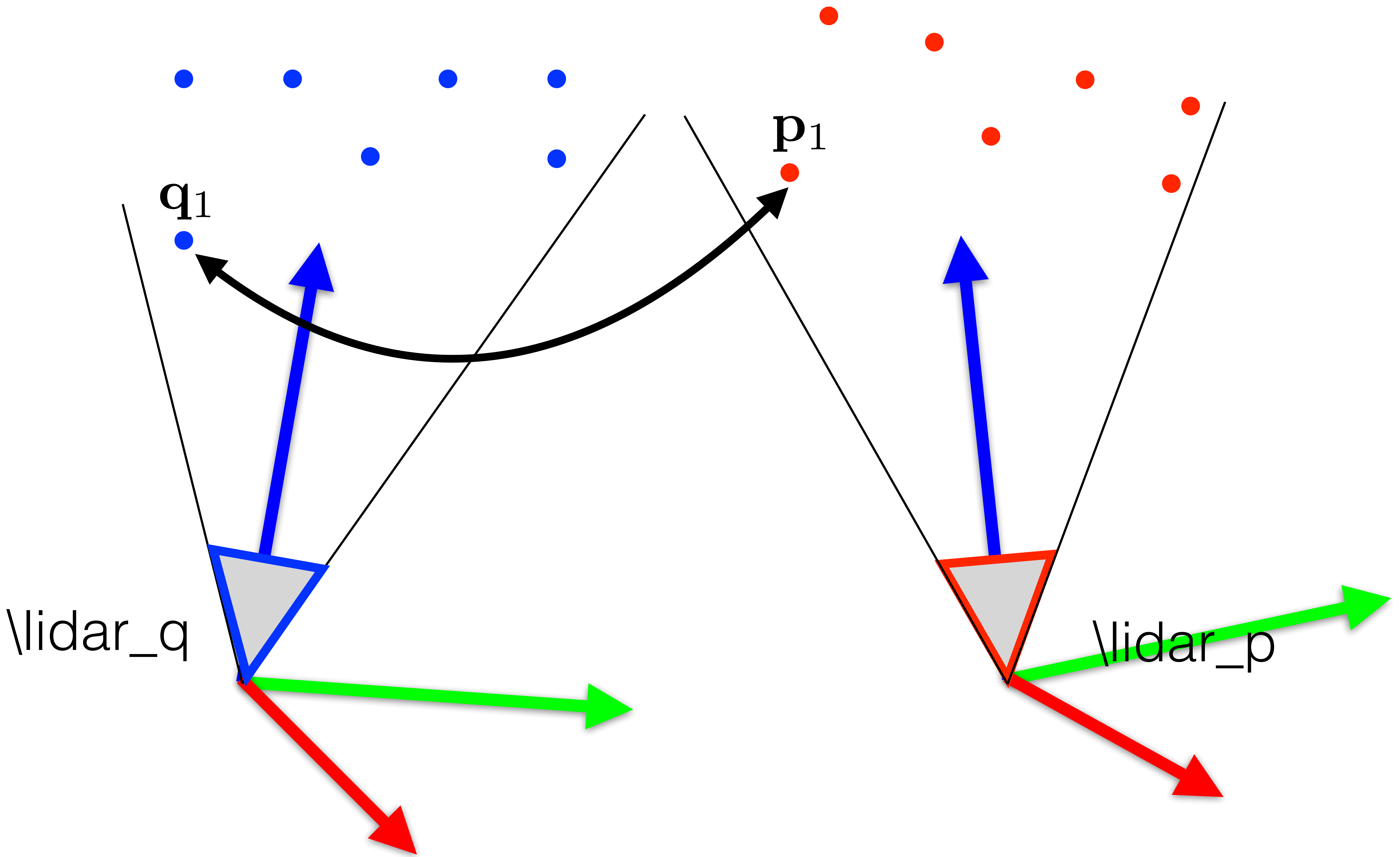


Mutual calibration of two coordinate frames



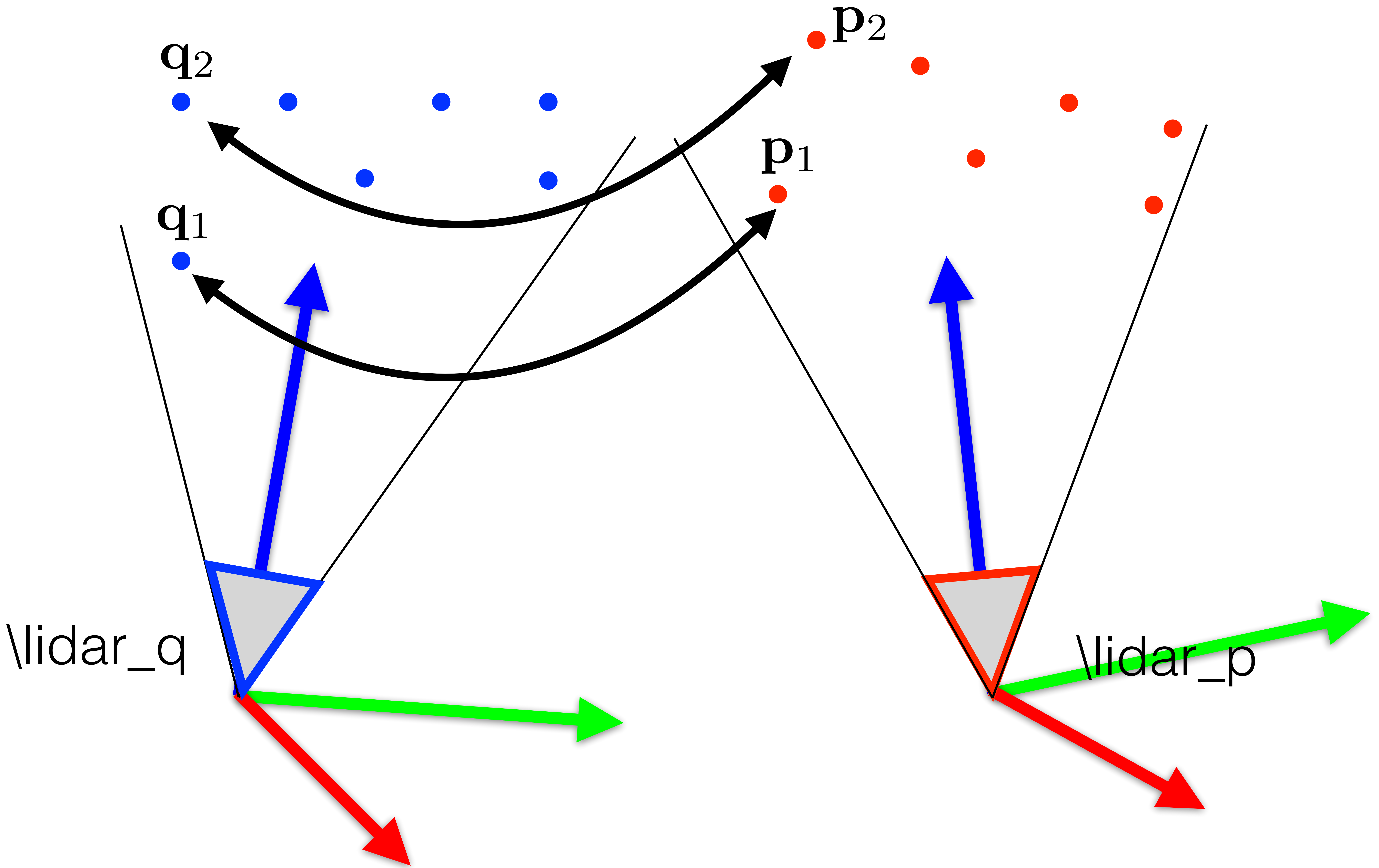
Mutual calibration of two coordinate frames

3D-3D correspondences

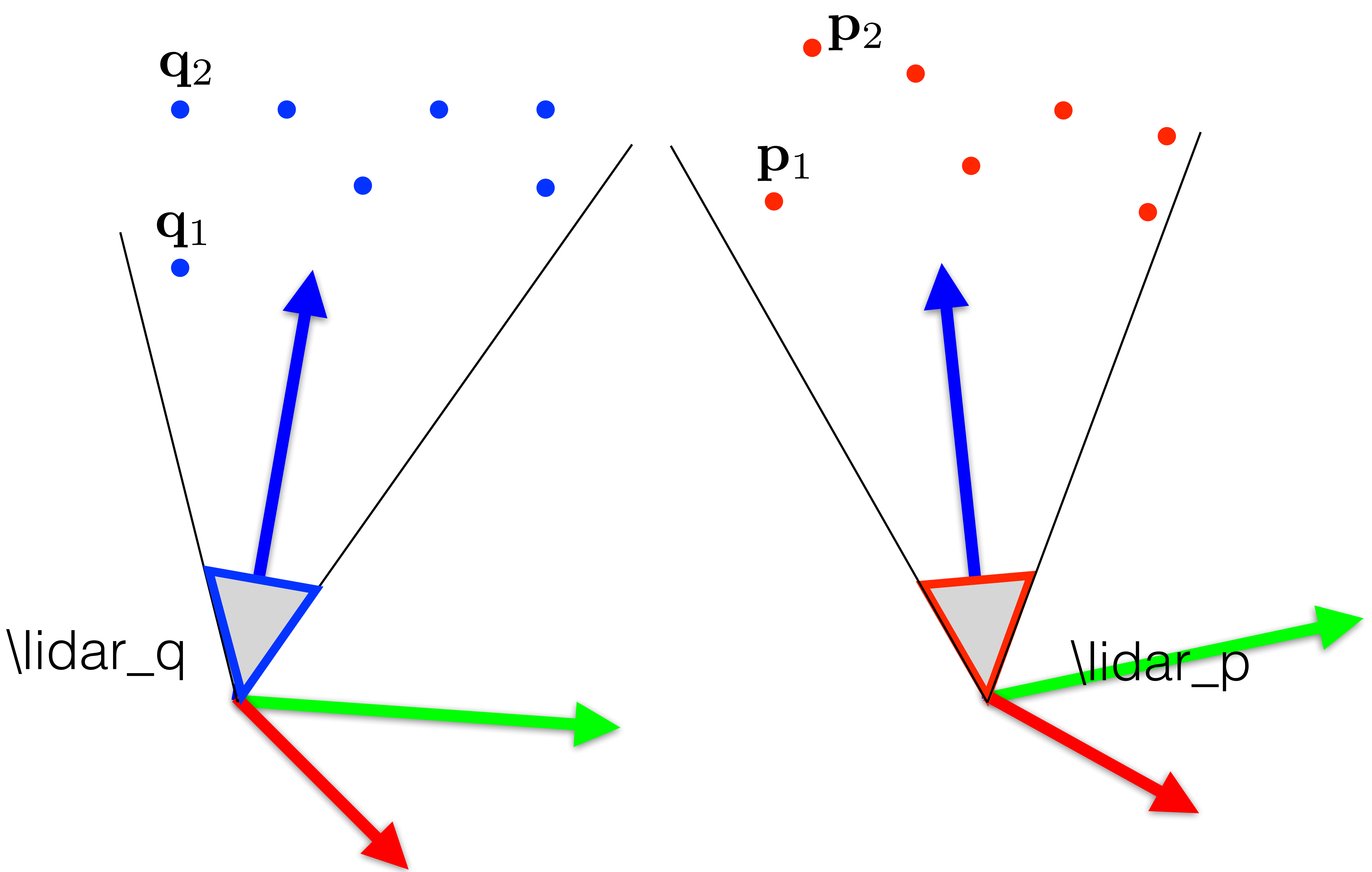


Mutual calibration of two coordinate frames

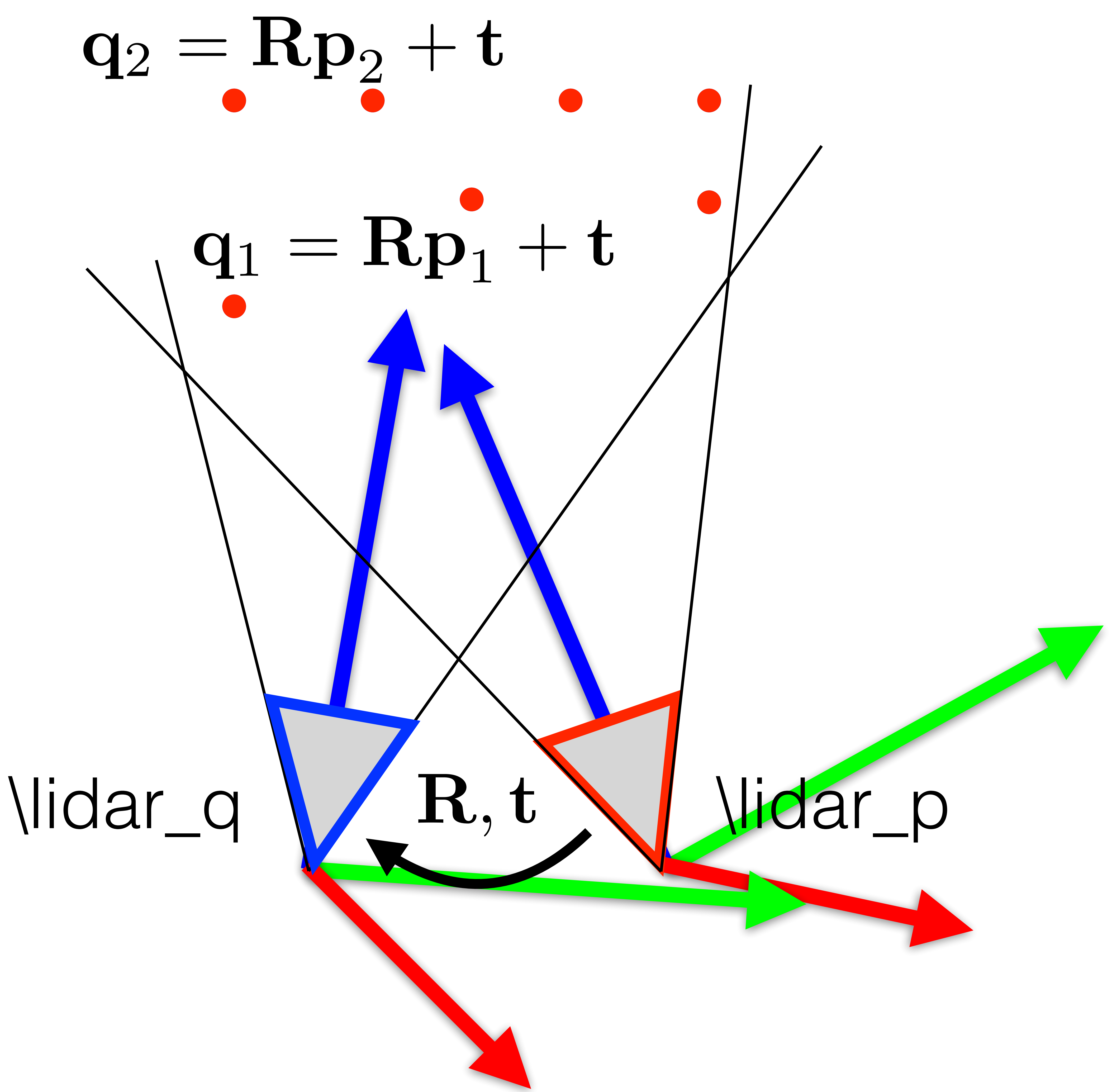
3D-3D correspondences



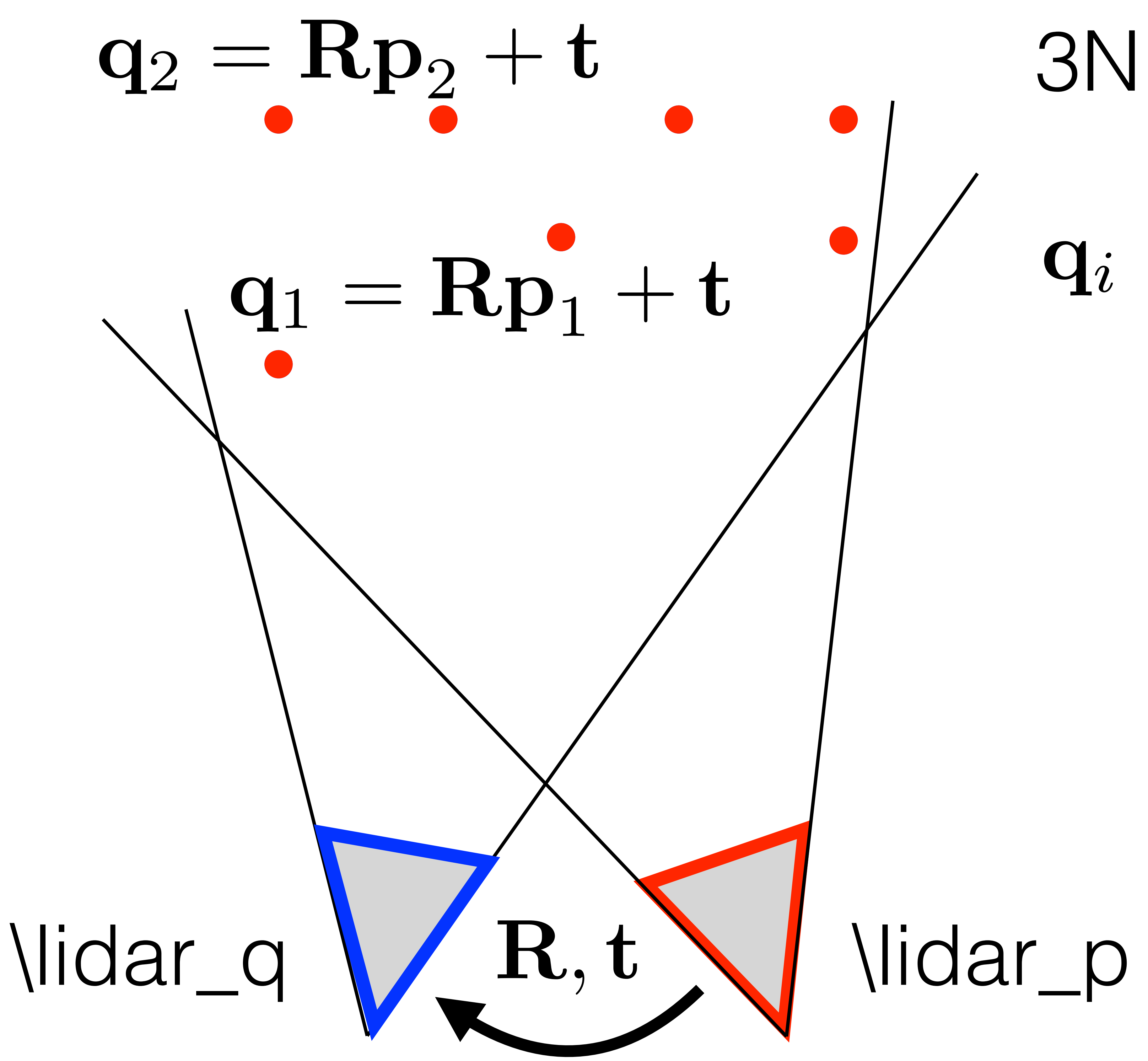
Mutual calibration of two coordinate frames



Mutual calibration of two coordinate frames



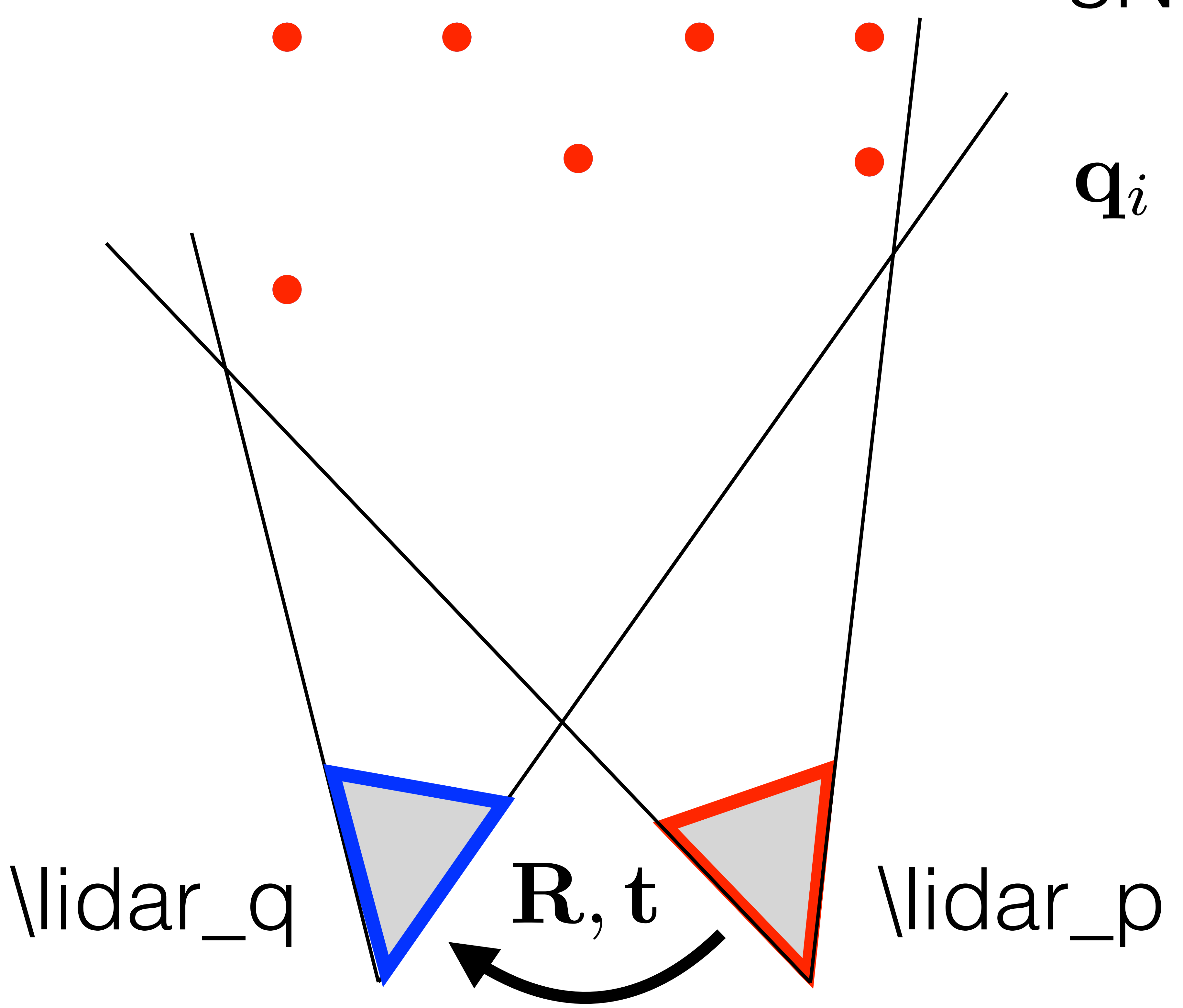
Mutual calibration of two coordinate frames



3N equations ($\mathbf{R}, \mathbf{t}$ unknowns):

$$\mathbf{q}_i = \mathbf{R}\mathbf{p}_i + \mathbf{t} \quad \forall_{i=1 \dots N}$$

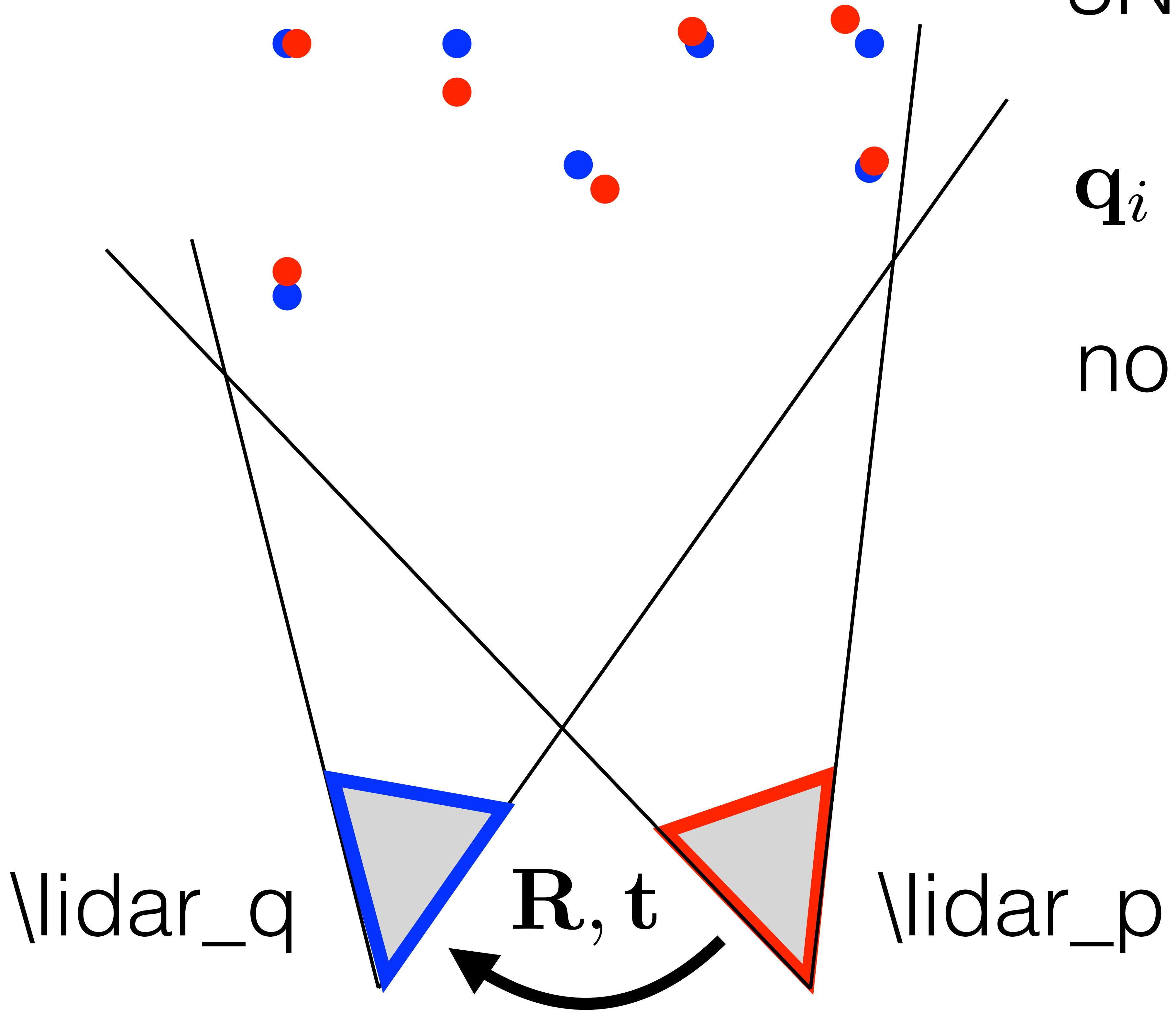
Mutual calibration of two coordinate frames



3N equations ($\mathbf{R}, \mathbf{t}$ unknowns):

$$\mathbf{q}_i = \mathbf{R}\mathbf{p}_i + \mathbf{t} \quad \forall_{i=1 \ldots N}$$

Mutual calibration of two coordinate frames

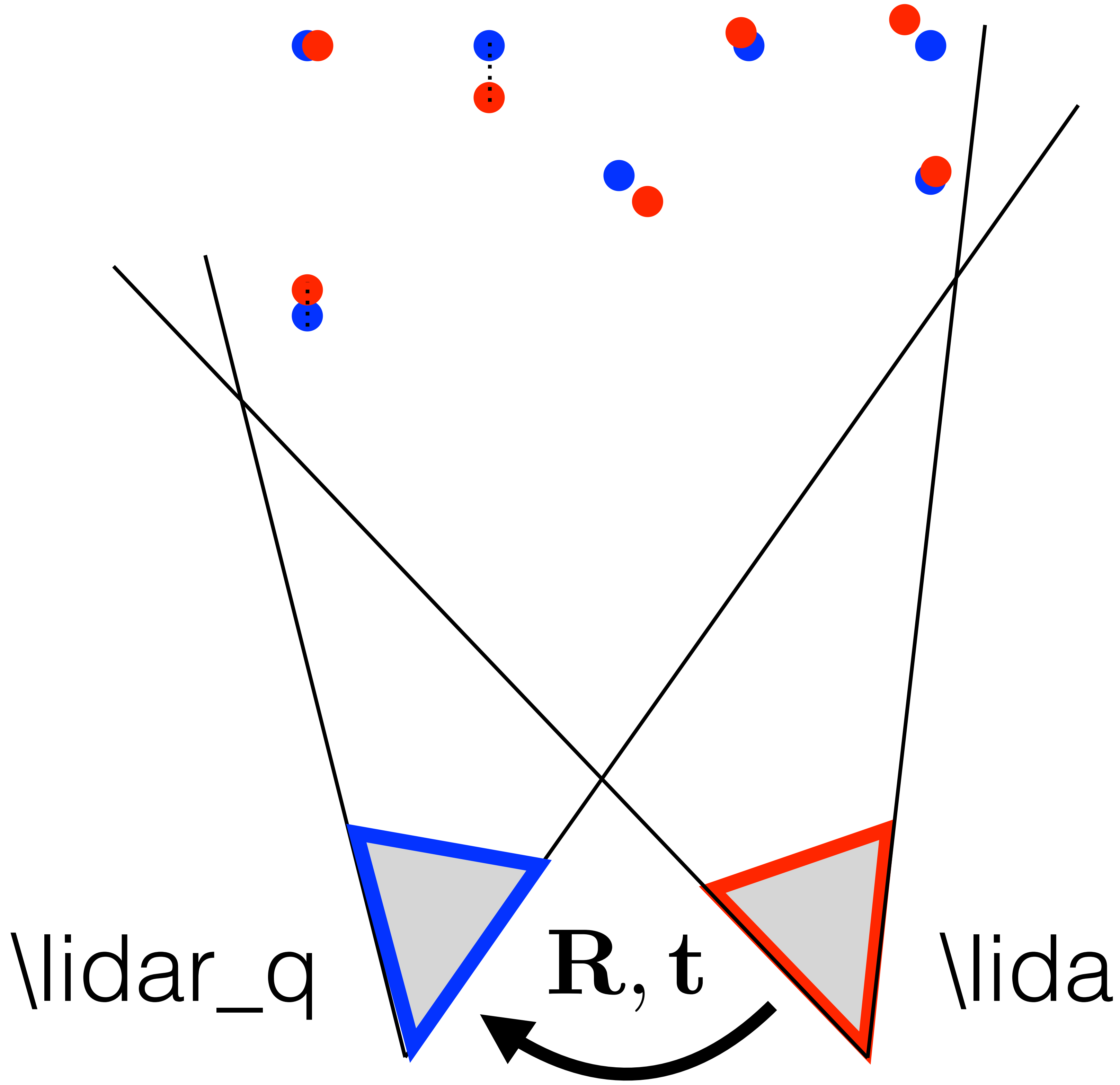


3N equations ($\mathbf{R}, \mathbf{t}$ unknowns):

$$\mathbf{q}_i \neq \mathbf{R}\mathbf{p}_i + \mathbf{t} \quad \forall_{i=1\ldots N}$$

noise => no exact solution

Mutual calibration of two coordinate frames



3N equations ($\mathbf{R}, \mathbf{t}$ unknowns):

$$\mathbf{q}_i \neq \mathbf{R}\mathbf{p}_i + \mathbf{t} \quad \forall_{i=1 \ldots N}$$

noise => no exact solution

Minimize sum of squared differences between left and right-hand side.

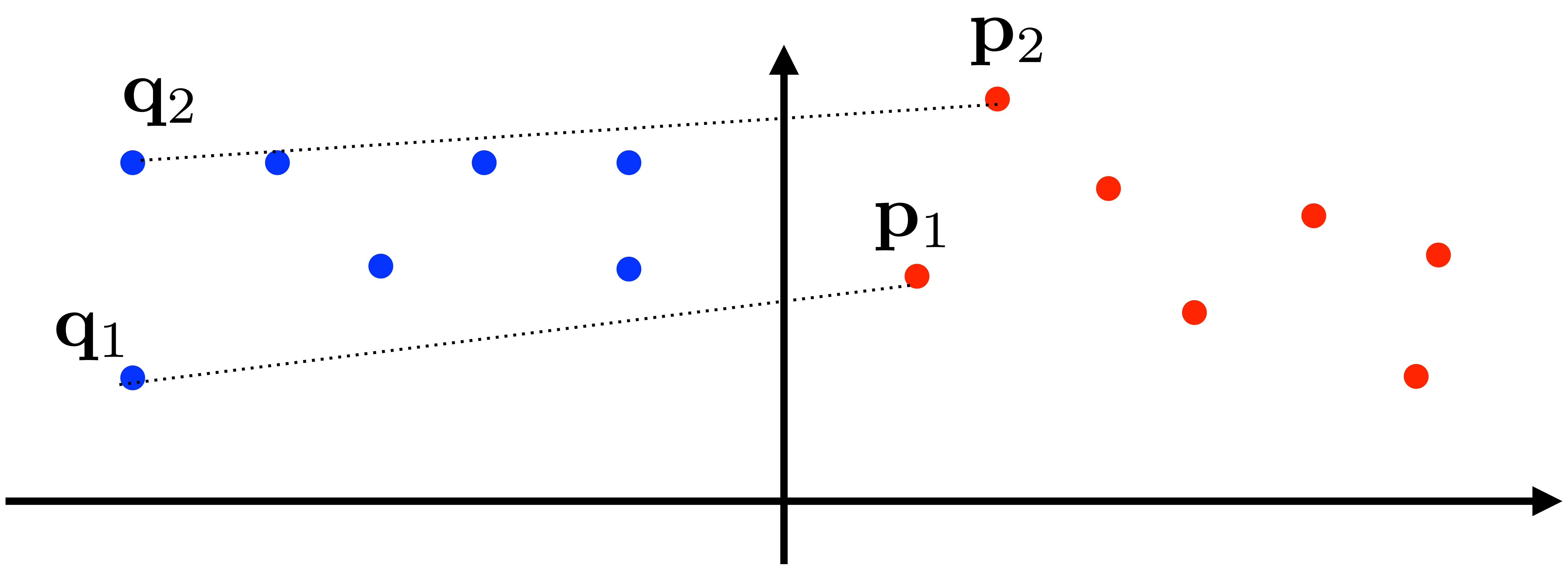
ML estimate wrt:
- gaussian noise
- i.i.d measurements

$$\mathbf{R}^*, \mathbf{t}^* = \underset{\mathbf{R} \in SO(3), \mathbf{t} \in \mathcal{R}^3}{\arg\min} \sum_i \|\mathbf{R}\mathbf{p}_i + \mathbf{t} - \mathbf{q}_i\|_2^2$$

Mutual calibration of two coordinate frames

$$\mathbf{R}^*, \mathbf{t}^* = \arg\min_{\mathbf{R} \in SO(3), \mathbf{t} \in \mathcal{R}^3} \sum_i \|\mathbf{R}\mathbf{p}_i + \mathbf{t} - \mathbf{q}_i\|_2^2$$

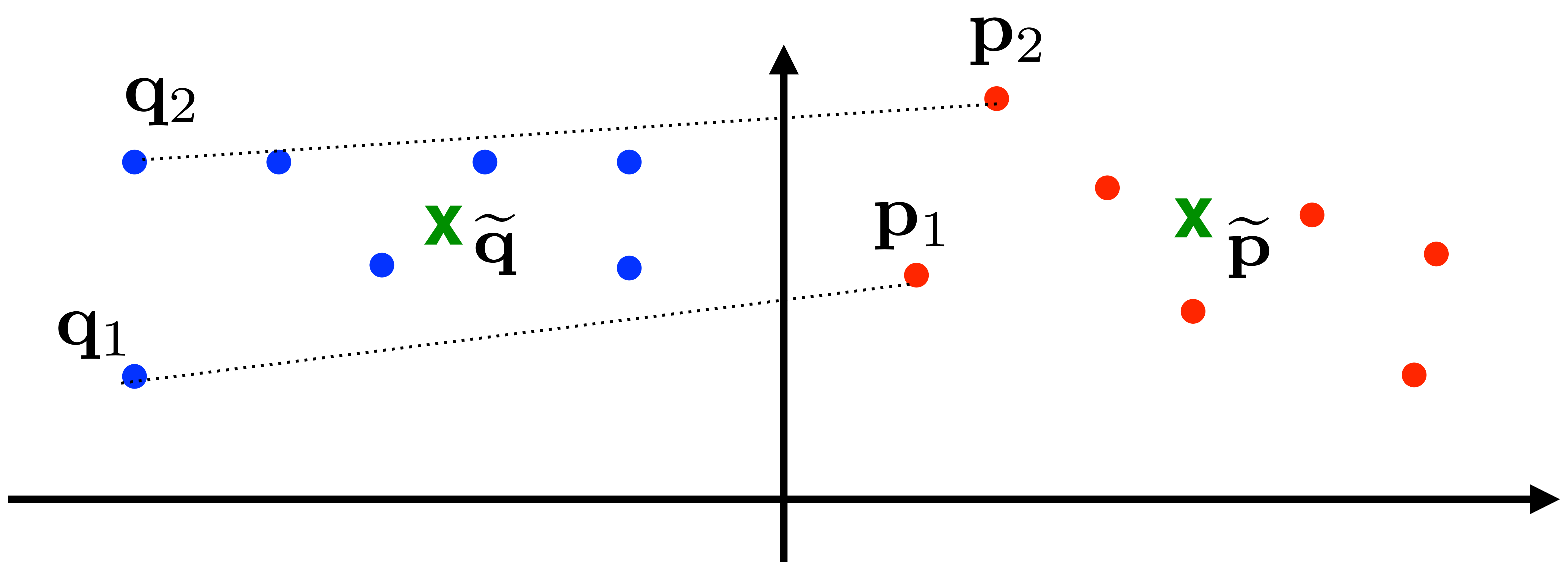
Solution:



Mutual calibration of two coordinate frames

$$\mathbf{R}^*, \mathbf{t}^* = \underset{\mathbf{R} \in SO(3), \mathbf{t} \in \mathcal{R}^3}{\arg\min} \sum_i \|\mathbf{R}\mathbf{p}_i + \mathbf{t} - \mathbf{q}_i\|_2^2$$

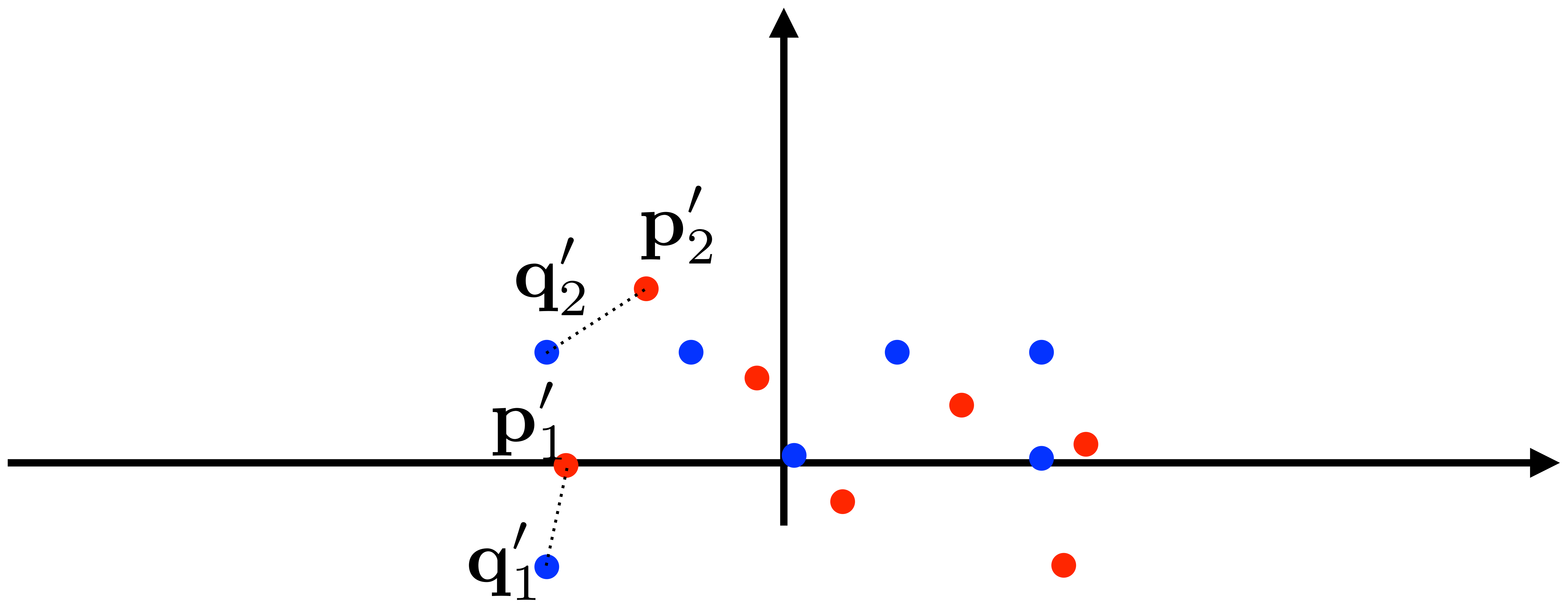
Solution: $\mathbf{p}'_i = \mathbf{p}_i - \underbrace{\frac{1}{N}\sum_i \mathbf{p}_i}_{\widetilde{\mathbf{p}}}, \quad \mathbf{q}'_i = \mathbf{q}_i - \underbrace{\frac{1}{N}\sum_i \mathbf{q}_i}_{\widetilde{\mathbf{q}}}$



Mutual calibration of two coordinate frames

$$\mathbf{R}^*, \mathbf{t}^* = \underset{\mathbf{R}\in SO(3), \mathbf{t}\in\mathcal{R}^3}{\arg\min} \sum_i \|\mathbf{R}\mathbf{p}_i + \mathbf{t} - \mathbf{q}_i\|_2^2$$

Solution: $$\mathbf{p}'_i = \mathbf{p}_i - \underbrace{\frac{1}{N}\sum_i \mathbf{p}_i}_{\widetilde{\mathbf{p}}}, \quad \mathbf{q}'_i = \mathbf{q}_i - \underbrace{\frac{1}{N}\sum_i \mathbf{q}_i}_{\widetilde{\mathbf{q}}}$$

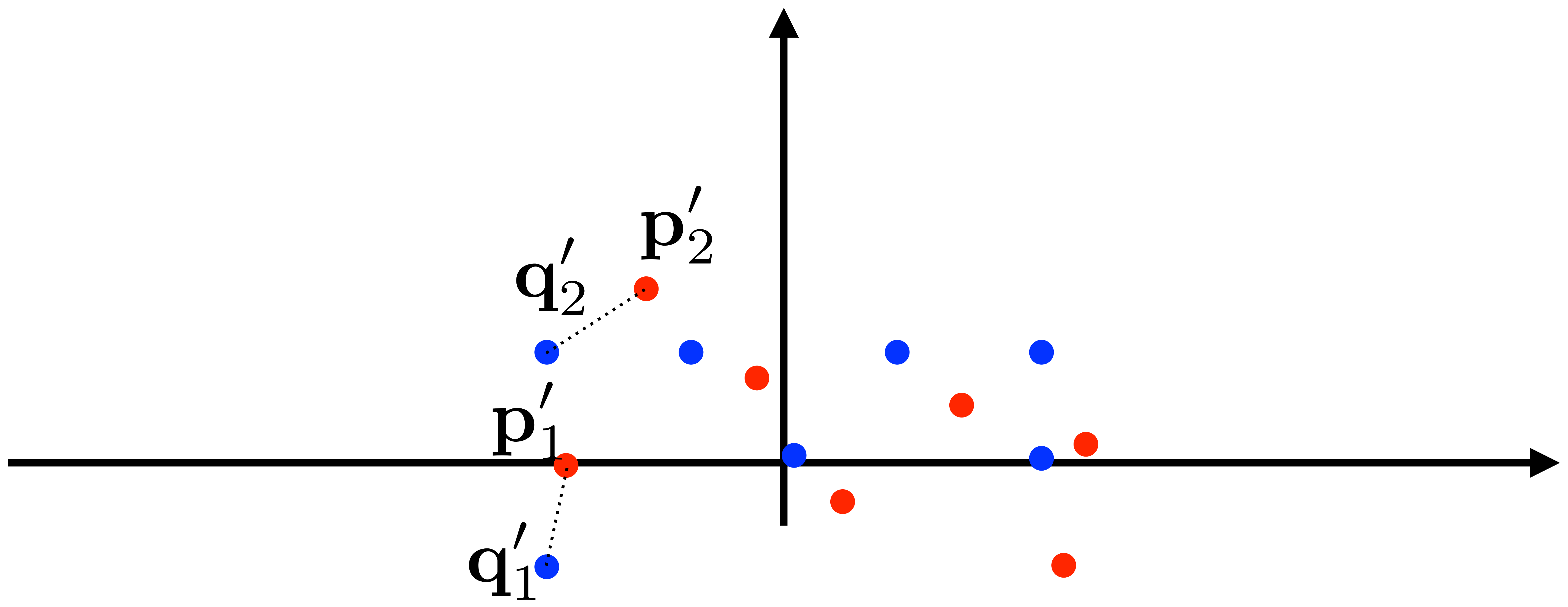


Mutual calibration of two coordinate frames

$$\mathbf{R}^*, \mathbf{t}^* = \underset{\mathbf{R}\in SO(3), \mathbf{t}\in\mathcal{R}^3}{\arg\min} \sum_i \|\mathbf{R}\mathbf{p}_i + \mathbf{t} - \mathbf{q}_i\|_2^2$$

$$= \underset{\mathbf{R}\in SO(3), \mathbf{t}\in\mathcal{R}^3}{\arg\min} \sum_i \|\mathbf{R}\mathbf{p}'_i - \mathbf{q}'_i\|_2^2 + \|\underbrace{\mathbf{R}\widetilde{\mathbf{p}} + \mathbf{t} - \widetilde{\mathbf{q}}}_{\mathbf{t}'}\|_2^2$$

Solution: $\mathbf{p}'_i = \mathbf{p}_i - \underbrace{\frac{1}{N}\sum_i \mathbf{p}_i}_{\widetilde{\mathbf{p}}}, \quad \mathbf{q}'_i = \mathbf{q}_i - \underbrace{\frac{1}{N}\sum_i \mathbf{q}_i}_{\widetilde{\mathbf{q}}}$



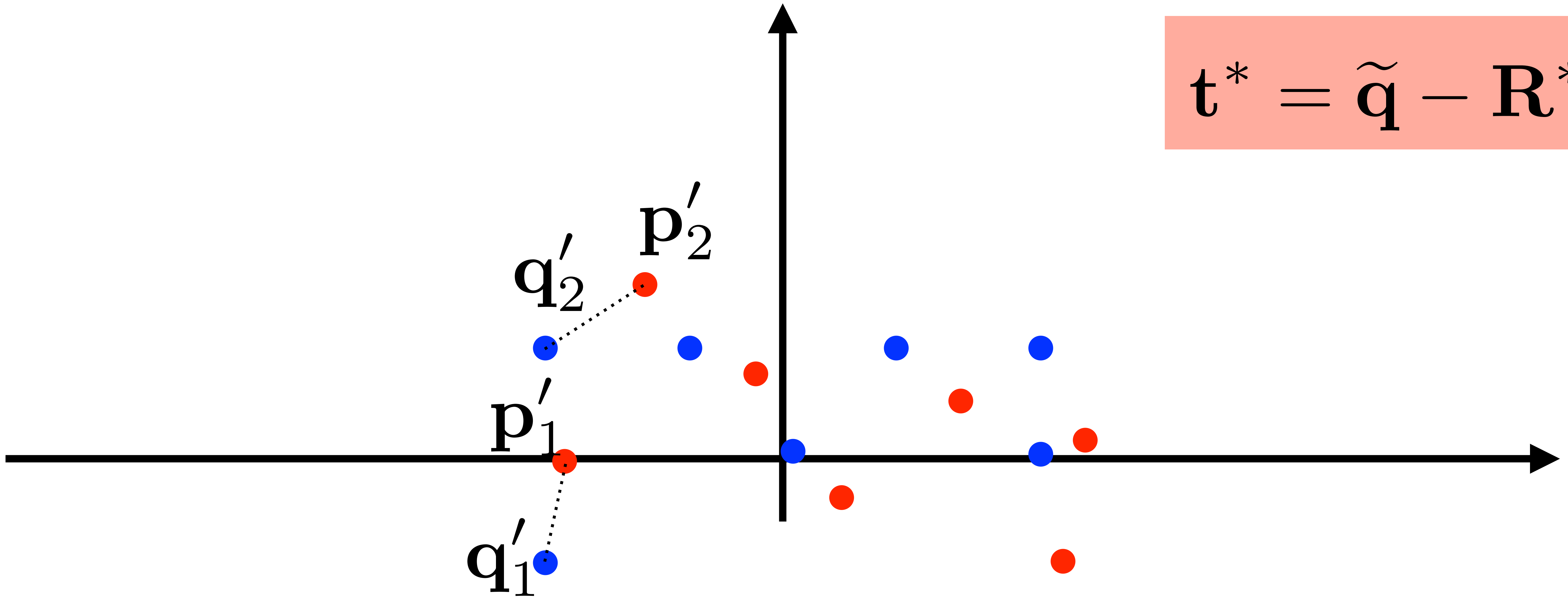
Mutual calibration of two coordinate frames

$$\mathbf{R}^*, \mathbf{t}^* = \arg\min_{\mathbf{R}\in SO(3), \mathbf{t}\in\mathcal{R}^3} \sum_i \|\mathbf{R}\mathbf{p}_i + \mathbf{t} - \mathbf{q}_i\|_2^2$$

$$= \arg\min_{\mathbf{R}\in SO(3), \mathbf{t}\in\mathcal{R}^3} \sum_i \|\mathbf{R}\mathbf{p}'_i - \mathbf{q}'_i\|_2^2 + \|\underbrace{\mathbf{R}\widetilde{\mathbf{p}} + \mathbf{t} - \widetilde{\mathbf{q}}}_{\mathbf{t}'}\|_2^2$$

Solution: $\mathbf{p}'_i = \mathbf{p}_i - \underbrace{\frac{1}{N}\sum_i \mathbf{p}_i}_{\widetilde{\mathbf{p}}}, \quad \mathbf{q}'_i = \mathbf{q}_i - \underbrace{\frac{1}{N}\sum_i \mathbf{q}_i}_{\widetilde{\mathbf{q}}}$

$$\mathbf{t}^* = \widetilde{\mathbf{q}} - \mathbf{R}^*\widetilde{\mathbf{p}}$$



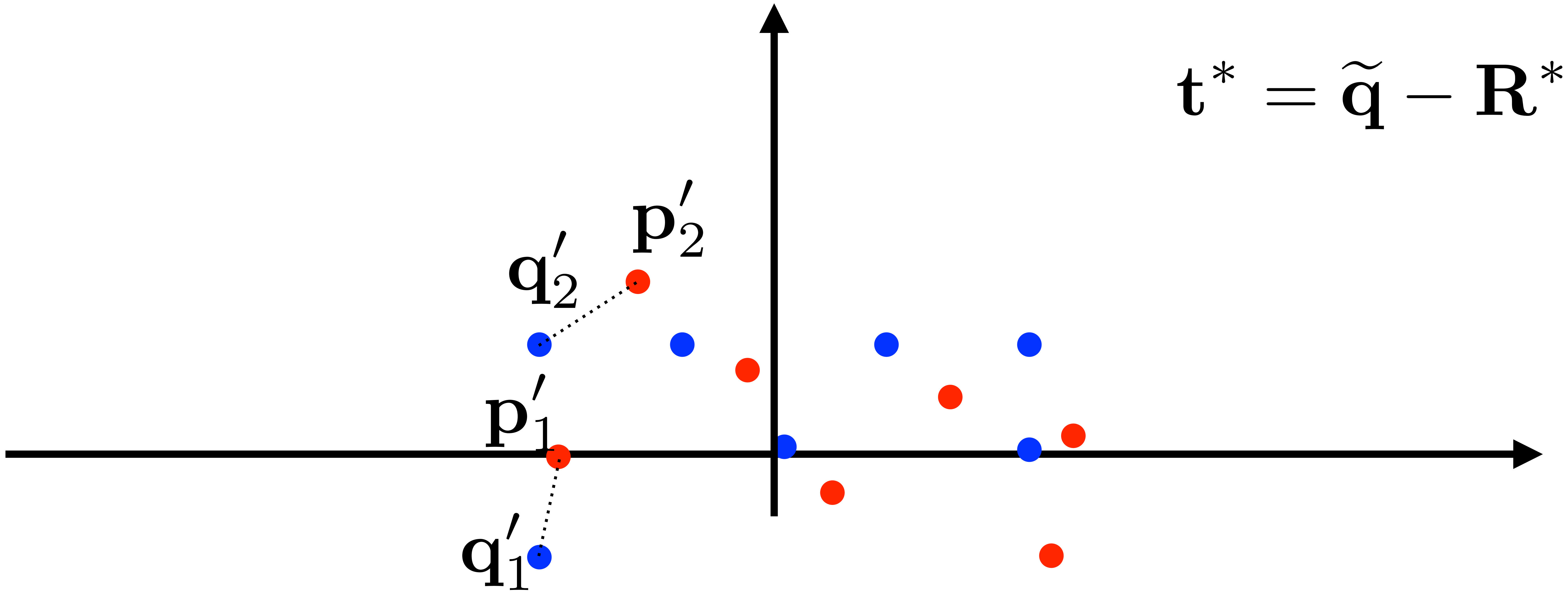
Mutual calibration of two coordinate frames

$$\mathbf{R}^*, \mathbf{t}^* = \underset{\mathbf{R} \in SO(3), \mathbf{t} \in \mathcal{R}^3}{\arg\min} \sum_i \|\mathbf{R}\mathbf{p}_i + \mathbf{t} - \mathbf{q}_i\|_2^2$$

$$= \underset{\mathbf{R} \in SO(3), \mathbf{t} \in \mathcal{R}^3}{\arg\min} \sum_i \|\mathbf{R}\mathbf{p}'_i - \mathbf{q}'_i\|_2^2 + \|\underbrace{\mathbf{R}\widetilde{\mathbf{p}} + \mathbf{t} - \widetilde{\mathbf{q}}}_{\mathbf{t}'}\|_2^2$$

Solution: estimate covariante matrix: $\mathbf{H} = \sum_i \mathbf{p}'_i \mathbf{q}'^{\top}_i$

$$\mathbf{t}^* = \widetilde{\mathbf{q}} - \mathbf{R}^* \widetilde{\mathbf{p}}$$



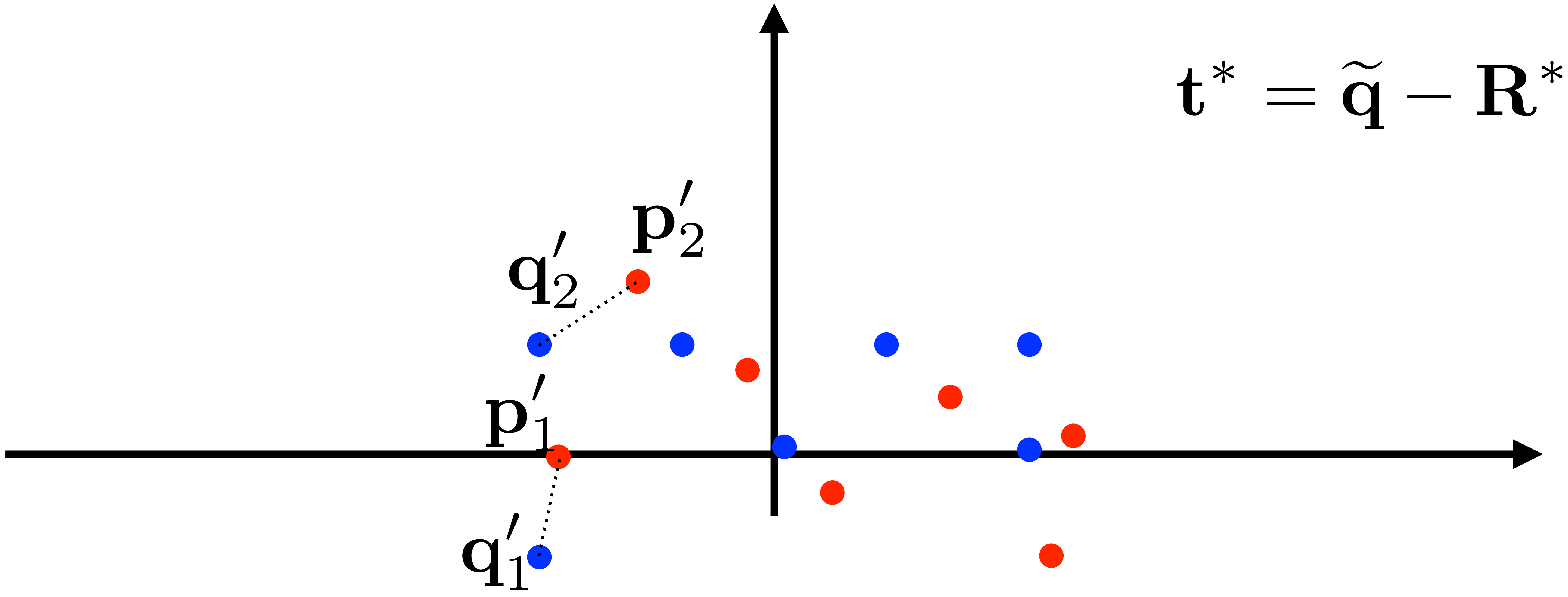
Mutual calibration of two coordinate frames

$$\mathbf{R}^*, \mathbf{t}^* = \underset{\mathbf{R}\in SO(3), \mathbf{t}\in\mathcal{R}^3}{\arg\min} \sum_i \|\mathbf{R}\mathbf{p}_i + \mathbf{t} - \mathbf{q}_i\|_2^2$$

$$= \underset{\mathbf{R}\in SO(3), \mathbf{t}\in\mathcal{R}^3}{\arg\min} \sum_i \|\mathbf{R}\mathbf{p}_i' - \mathbf{q}_i'\|_2^2 + \|\underbrace{\mathbf{R}\widetilde{\mathbf{p}} + \mathbf{t} - \widetilde{\mathbf{q}}}_{\mathbf{t}'}\|_2^2$$

Solution: estimate covariante matrix: $\mathbf{H} = \sum_i \mathbf{p}_i' \mathbf{q}_i'^\top$

find SVD decomposition: $\mathbf{H} = \mathbf{U}\mathbf{S}\mathbf{V}^\top$



$$\mathbf{t}^* = \widetilde{\mathbf{q}} - \mathbf{R}^*\widetilde{\mathbf{p}}$$

Mutual calibration of two coordinate frames

$$\mathbf{R}^*, \mathbf{t}^* = \arg\min_{\mathbf{R}\in SO(3),\mathbf{t}\in\mathcal{R}^3} \sum_i \|\mathbf{R}\mathbf{p}_i + \mathbf{t} - \mathbf{q}_i\|_2^2$$

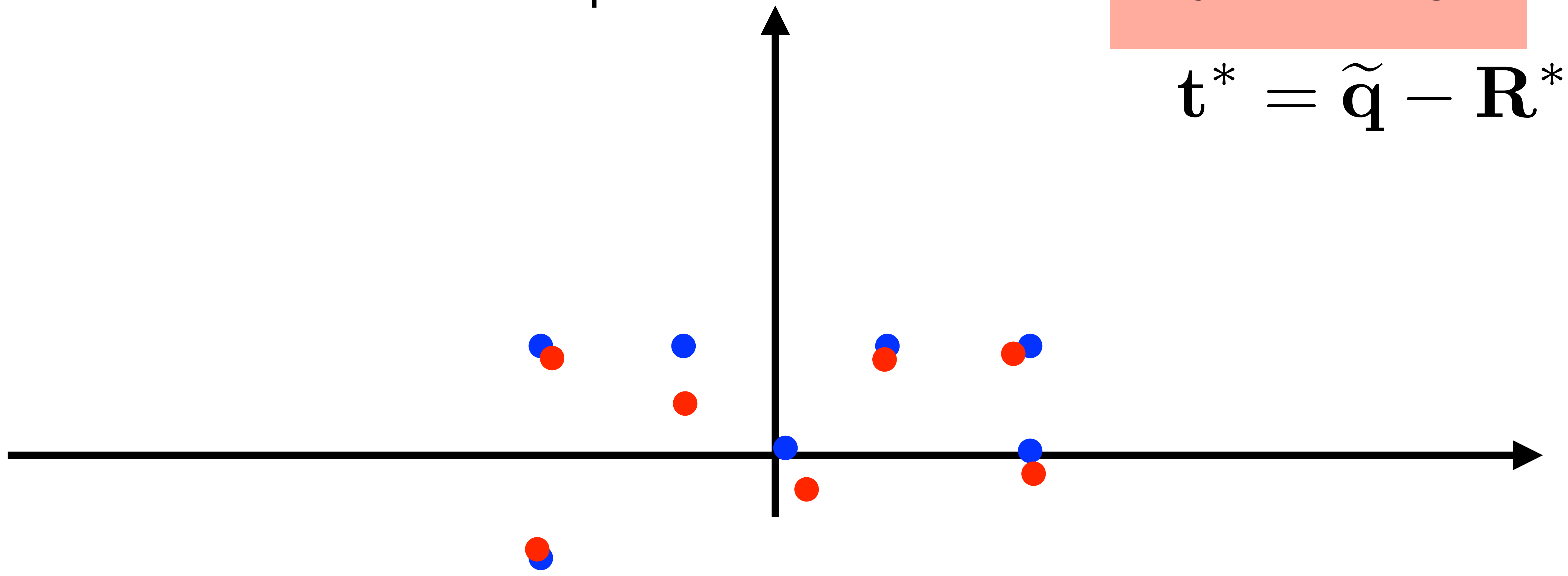
$$= \arg\min_{\mathbf{R}\in SO(3),\mathbf{t}\in\mathcal{R}^3} \sum_i \|\mathbf{R}\mathbf{p}'_i - \mathbf{q}'_i\|_2^2 + \|\underbrace{\mathbf{R}\widetilde{\mathbf{p}} + \mathbf{t} - \widetilde{\mathbf{q}}}_{\mathbf{t}'}\|_2^2$$

Solution: estimate covariante matrix: $\mathbf{H} = \sum_i \mathbf{p}'_i \mathbf{q}'^{\top}_i$

find SVD decomposition: $\mathbf{H} = \mathbf{U}\mathbf{S}\mathbf{V}^\top$

estimate optimal rotation: $\mathbf{R}^* = \mathbf{V}\mathbf{U}^\top$

$$\mathbf{t}^* = \widetilde{\mathbf{q}} - \mathbf{R}^*\widetilde{\mathbf{p}}$$



Mutual calibration of two coordinate frames

$$\mathbf{R}^*, \mathbf{t}^* = \arg\min_{\mathbf{R}\in SO(3), \mathbf{t}\in\mathcal{R}^3} \sum_i \|\mathbf{R}\mathbf{p}_i + \mathbf{t} - \mathbf{q}_i\|_2^2$$

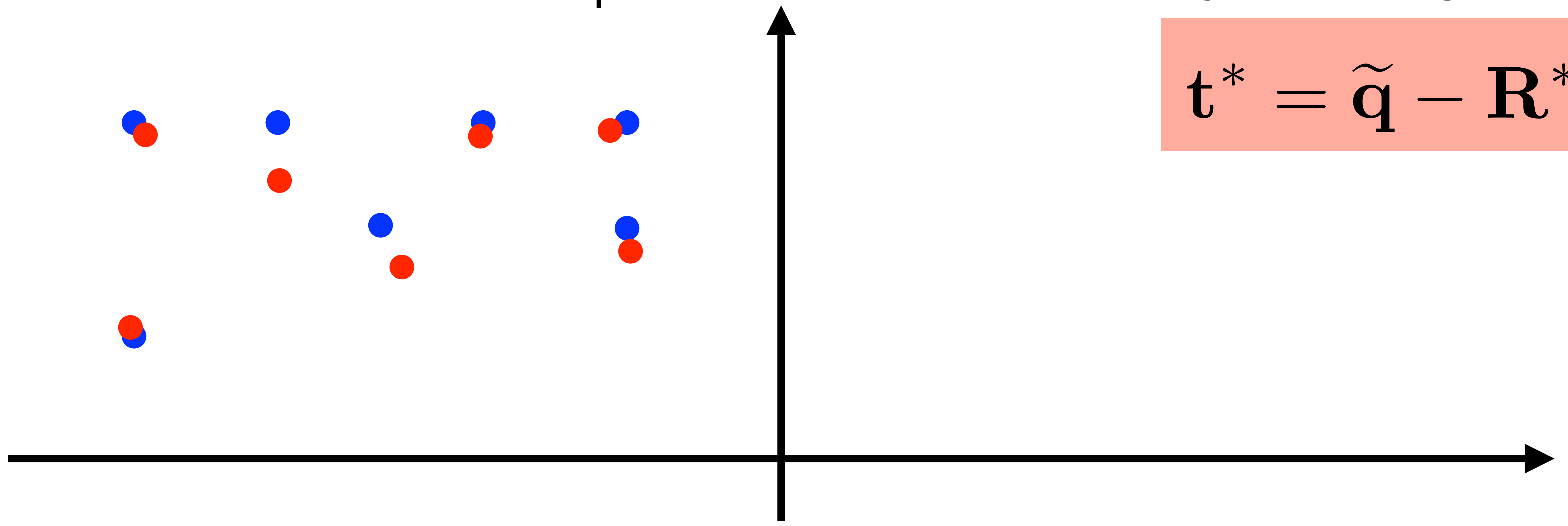
$$= \arg\min_{\mathbf{R}\in SO(3), \mathbf{t}\in\mathcal{R}^3} \sum_i \|\mathbf{R}\mathbf{p}'_i - \mathbf{q}'_i\|_2^2 + \|\underbrace{\mathbf{R}\widetilde{\mathbf{p}} + \mathbf{t} - \widetilde{\mathbf{q}}}_{\mathbf{t}'}\|_2^2$$

Solution: estimate covariante matrix: $\mathbf{H} = \sum_i \mathbf{p}'_i \mathbf{q}'^{\top}_i$

find SVD decomposition: $\mathbf{H} = \mathbf{U}\mathbf{S}\mathbf{V}^\top$

estimate optimal rotation: $\mathbf{R}^* = \mathbf{V}\mathbf{U}^\top$

$$\mathbf{t}^* = \widetilde{\mathbf{q}} - \mathbf{R}^*\widetilde{\mathbf{p}}$$



Mutual calibration of two coordinate frames

$$\mathbf{R}^*, \mathbf{t}^* = \arg\min_{\mathbf{R}\in SO(3),\mathbf{t}\in\mathcal{R}^3} \sum_i \|\mathbf{R}\mathbf{p}_i + \mathbf{t} - \mathbf{q}_i\|_2^2$$

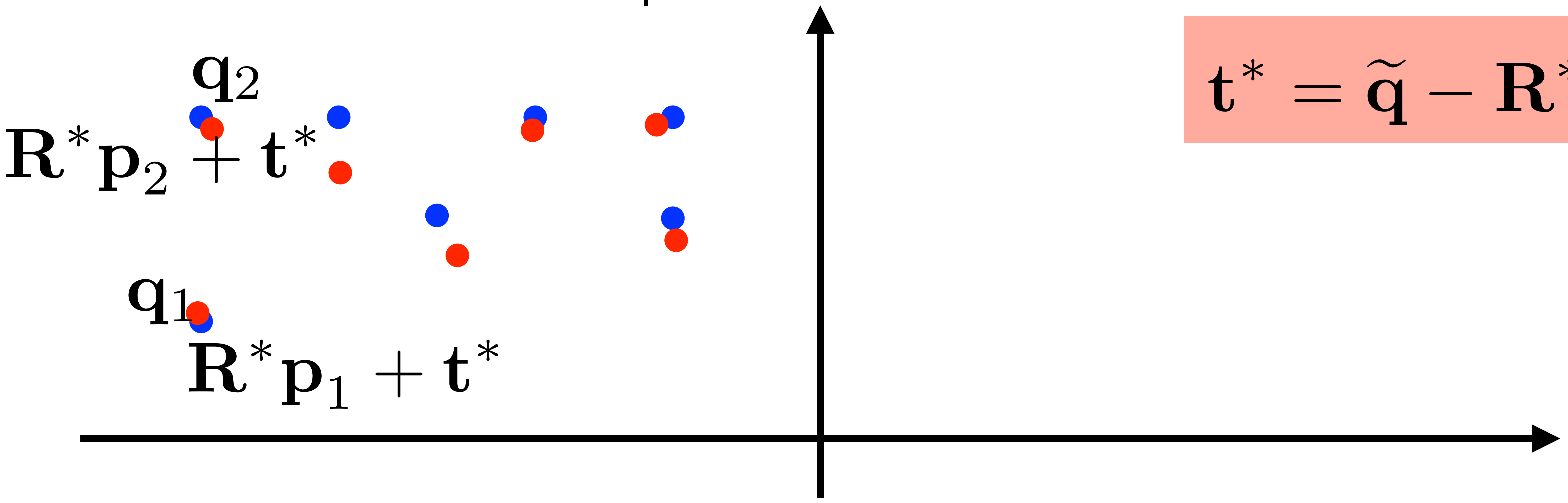
$$= \arg\min_{\mathbf{R}\in SO(3),\mathbf{t}\in\mathcal{R}^3} \sum_i \|\mathbf{R}\mathbf{p}'_i - \mathbf{q}'_i\|_2^2 + \|\underbrace{\mathbf{R}\widetilde{\mathbf{p}} + \mathbf{t} - \widetilde{\mathbf{q}}}_{\mathbf{t}'}\|_2^2$$

Solution: estimate covariante matrix: $\mathbf{H} = \sum_i \mathbf{p}'_i \mathbf{q}'^{\top}_i$

find SVD decomposition: $\mathbf{H} = \mathbf{U}\mathbf{S}\mathbf{V}^\top$

estimate optimal rotation: $\mathbf{R}^* = \mathbf{V}\mathbf{U}^\top$

$\mathbf{t}^* = \widetilde{\mathbf{q}} - \mathbf{R}^*\widetilde{\mathbf{p}}$



Mutual calibration of two coordinate frames

(1) Record pointclouds and manually estimate 3D-3D correspondences

(2) Solve: $$\mathbf{R}^*, \mathbf{t}^* = \arg\min_{\mathbf{R}\in SO(3), \mathbf{t}\in\mathcal{R}^3} \sum_i \|\mathbf{R}\mathbf{p}_i + \mathbf{t} - \mathbf{q}_i\|_2^2$$

Solution: $$\mathbf{R}^* = \mathbf{V}\mathbf{U}^\top$$
$$\mathbf{t}^* = \widetilde{\mathbf{q}} - \mathbf{R}^*\widetilde{\mathbf{p}}$$

In python:

```
H = P @ Q.T
U, S, V = np.linalg.svd(H, full_matrices=True)
```

Broadcasting static transformation between two c.f. in ROS:

```
broadcaster = tf2_ros.StaticTransformBroadcaster()
transform = geometry_msgs.msg.TransformStamped()
# compute transform from 3D-3D correspondences
broadcaster.sendTransform(transform)
```

Summary

- Robot consist of many distributed components (sensors, actuators, joints), each operating in its own dynamically evolving coordinate frame
- Transformation between two different poinclouds corresponding to a single rigid body is Euclidean motion (i.e. rotation R and translation t)
- Given 3D-3D correspondences, globally optimal alignment in L2 has closed-form solution (i.e. least-squares solution constrained SE(3) manifold)
    - Application in Robotics for SLAM.
    - Application in Computer graphics for alignment of 3D models
- Next: ICP SLAM

Proof [Arun-TPAMI-87]

$$\mathbf{R}^*, \mathbf{t}^* = \arg\min_{\mathbf{R}\in SO(3), \mathbf{t}\in\mathcal{R}^3} \sum_i \|\mathbf{R}\mathbf{p}_i + \mathbf{t} - \mathbf{q}_i\|_2^2 =$$

Proof [Arun-TPAMI-87]

$$\mathbf{R}^*, \mathbf{t}^* = \arg\min_{\mathbf{R}\in SO(3), \mathbf{t}\in\mathcal{R}^3} \sum_i \|\mathbf{R}\mathbf{p}_i + \mathbf{t} - \mathbf{q}_i\|_2^2 =$$

$$= \arg\min_{\mathbf{R}\in SO(3), \mathbf{t}\in\mathcal{R}^3} \sum_i \|\mathbf{R}(\mathbf{p}'_i + \widetilde{\mathbf{p}}) + \mathbf{t} - \mathbf{q}'_i - \widetilde{\mathbf{q}}\|_2^2 =$$

Proof [Arun-TPAMI-87]

$$\mathbf{R}^*, \mathbf{t}^* = \arg\min_{\mathbf{R} \in SO(3), \mathbf{t} \in \mathcal{R}^3} \sum_i \|\mathbf{R}\mathbf{p}_i + \mathbf{t} - \mathbf{q}_i\|_2^2 =$$

$$= \arg\min_{\mathbf{R} \in SO(3), \mathbf{t} \in \mathcal{R}^3} \sum_i \|\mathbf{R}(\mathbf{p}_i' + \widetilde{\mathbf{p}}) + \mathbf{t} - \mathbf{q}_i' - \widetilde{\mathbf{q}}\|_2^2 =$$

$$= \arg\min_{\mathbf{R} \in SO(3), \mathbf{t} \in \mathcal{R}^3} \sum_i \|\mathbf{R}\mathbf{p}_i' - \mathbf{q}_i' + \underbrace{\mathbf{R}\widetilde{\mathbf{p}} + \mathbf{t} - \widetilde{\mathbf{q}}}_{\mathbf{t}'}\|_2^2 =$$

Proof [Arun-TPAMI-87]

$$\mathbf{R}^*, \mathbf{t}^* = \underset{\mathbf{R} \in SO(3), \mathbf{t} \in \mathcal{R}^3}{\arg\min} \sum_i \|\mathbf{R}\mathbf{p}_i + \mathbf{t} - \mathbf{q}_i\|_2^2 =$$

$$= \underset{\mathbf{R} \in SO(3), \mathbf{t} \in \mathcal{R}^3}{\arg\min} \sum_i \|\mathbf{R}(\mathbf{p}'_i + \widetilde{\mathbf{p}}) + \mathbf{t} - \mathbf{q}'_i - \widetilde{\mathbf{q}}\|_2^2 =$$

$$= \underset{\mathbf{R} \in SO(3), \mathbf{t} \in \mathcal{R}^3}{\arg\min} \sum_i \|\mathbf{R}\mathbf{p}'_i - \mathbf{q}'_i + \underbrace{\mathbf{R}\widetilde{\mathbf{p}} + \mathbf{t} - \widetilde{\mathbf{q}}}_{\mathbf{t}'}\|_2^2 =$$

$$= \underset{\mathbf{R} \in SO(3), \mathbf{t} \in \mathcal{R}^3}{\arg\min} \sum_i (\mathbf{R}\mathbf{p}'_i - \mathbf{q}'_i + \mathbf{t}')^\top (\mathbf{R}\mathbf{p}'_i - \mathbf{q}'_i + \mathbf{t}') =$$

Proof [Arun-TPAMI-87]

$$\mathbf{R}^*, \mathbf{t}^* = \arg\min_{\mathbf{R}\in SO(3),\mathbf{t}\in\mathcal{R}^3} \sum_i \|\mathbf{R}\mathbf{p}_i + \mathbf{t} - \mathbf{q}_i\|_2^2 =$$

$$= \arg\min_{\mathbf{R}\in SO(3),\mathbf{t}\in\mathcal{R}^3} \sum_i \|\mathbf{R}(\mathbf{p}_i' + \widetilde{\mathbf{p}}) + \mathbf{t} - \mathbf{q}_i' - \widetilde{\mathbf{q}}\|_2^2 =$$

$$= \arg\min_{\mathbf{R}\in SO(3),\mathbf{t}\in\mathcal{R}^3} \sum_i \|\mathbf{R}\mathbf{p}_i' - \mathbf{q}_i' + \underbrace{\mathbf{R}\widetilde{\mathbf{p}} + \mathbf{t} - \widetilde{\mathbf{q}}}_{\mathbf{t}'}\|_2^2 =$$

$$= \arg\min_{\mathbf{R}\in SO(3),\mathbf{t}\in\mathcal{R}^3} \sum_i (\mathbf{R}\mathbf{p}_i' - \mathbf{q}_i' + \mathbf{t}')^\top(\mathbf{R}\mathbf{p}_i' - \mathbf{q}_i' + \mathbf{t}') =$$

$$= \arg\min_{\mathbf{R}\in SO(3),\mathbf{t}\in\mathcal{R}^3} \sum_i \|\mathbf{R}\mathbf{p}_i' - \mathbf{q}_i'\|_2^2 + \underbrace{\sum_i 2(\mathbf{R}\mathbf{p}_i' - \mathbf{q}_i')\mathbf{t}'}_{=0} + \|\mathbf{t}'\|_2^2 =$$

Proof [Arun-TPAMI-87]

$$\mathbf{R}^*, \mathbf{t}^* = \arg\min_{\mathbf{R}\in SO(3), \mathbf{t}\in\mathcal{R}^3} \sum_i \|\mathbf{R}\mathbf{p}_i + \mathbf{t} - \mathbf{q}_i\|_2^2 =$$

$$= \arg\min_{\mathbf{R}\in SO(3), \mathbf{t}\in\mathcal{R}^3} \sum_i \|\mathbf{R}(\mathbf{p}'_i + \widetilde{\mathbf{p}}) + \mathbf{t} - \mathbf{q}'_i - \widetilde{\mathbf{q}}\|_2^2 =$$

$$= \arg\min_{\mathbf{R}\in SO(3), \mathbf{t}\in\mathcal{R}^3} \sum_i \|\mathbf{R}\mathbf{p}'_i - \mathbf{q}'_i + \underbrace{\mathbf{R}\widetilde{\mathbf{p}} + \mathbf{t} - \widetilde{\mathbf{q}}}_{\mathbf{t}'}\|_2^2 =$$

$$= \arg\min_{\mathbf{R}\in SO(3), \mathbf{t}\in\mathcal{R}^3} \sum_i (\mathbf{R}\mathbf{p}'_i - \mathbf{q}'_i + \mathbf{t}')^\top (\mathbf{R}\mathbf{p}'_i - \mathbf{q}'_i + \mathbf{t}') =$$

$$= \arg\min_{\mathbf{R}\in SO(3), \mathbf{t}\in\mathcal{R}^3} \sum_i \|\mathbf{R}\mathbf{p}'_i - \mathbf{q}'_i\|_2^2 + \underbrace{\sum_i 2(\mathbf{R}\mathbf{p}'_i - \mathbf{q}'_i)\mathbf{t}'}_{=0} + \|\mathbf{t}'\|_2^2 =$$

$$= \arg\min_{\mathbf{R}\in SO(3), \mathbf{t}\in\mathcal{R}^3} \sum_i \|\mathbf{R}\mathbf{p}'_i - \mathbf{q}'_i\|_2^2 + \|\mathbf{t}'\|_2^2$$

we can reach second term zero by $\mathbf{t} = \widetilde{\mathbf{q}} - \mathbf{R}\widetilde{\mathbf{p}} = \mathbf{t}^*$

Proof [Arun-TPAMI-87]

$$= \arg\min_{\mathbf{R}\in SO(3),\mathbf{t}\in\mathcal{R}^3} \sum_i \|\mathbf{R}\mathbf{p}_i' - \mathbf{q}_i'\|_2^2 + \|\mathbf{t}'\|_2^2$$

we can reach second term zero by $\mathbf{t} = \tilde{\mathbf{q}} - \mathbf{R}\tilde{\mathbf{p}} = \mathbf{t}^*$

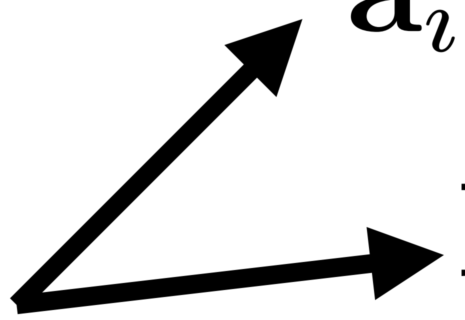
$$\arg\min_{\mathbf{R}\in SO(3)} \sum_i \|\mathbf{R}\mathbf{p}_i' - \mathbf{q}_i'\|_2^2 = \arg\max_{\mathbf{R}\in SO(3)} \sum_i \mathbf{q}_i'^\top \mathbf{R}\mathbf{p}_i' =$$

$$= \arg\max_{\mathbf{R}\in SO(3)} \sum_i \underbrace{\mathbf{q}_i'^\top}_{\mathbf{a}_i} \underbrace{\mathbf{R}\mathbf{p}_i'}_{\mathbf{b}_i} = \arg\max_{\mathbf{R}\in SO(3)} \ \text{trace}\ \mathbf{R}\underbrace{\mathbf{P}\mathbf{Q}^\top}_{\mathbf{H}} = \mathbf{V}\mathbf{U}^\top$$

$$\arg\max_{\mathbf{R}',\mathbf{R}^*\in SO(3)} \ \text{trace}\ \mathbf{R}'\mathbf{R}^*\mathbf{U}\mathbf{S}\mathbf{V}^\top$$ … expand into two rotations

$$\arg\max_{\mathbf{R}'\in SO(3)} \text{trace}\ \mathbf{R}' \underbrace{\mathbf{V}\mathbf{U}^\top}_{\mathbf{R}^*} \underbrace{\mathbf{U}\mathbf{S}\mathbf{V}^\top}_{\mathbf{H}} = \arg\max_{\mathbf{R}'\in SO(3)} \text{trace}\ \mathbf{R}' \underbrace{(\mathbf{V}\sqrt{\mathbf{S}})}_{\mathbf{A}} \underbrace{(\sqrt{\mathbf{S}}\mathbf{V})^\top}_{\mathbf{A}^\top} =$$

$$= \arg\max_{\mathbf{R}'\in SO(3)} \sum_i \mathbf{a}_i^\top \mathbf{R}'\mathbf{a}_i = \mathbf{E}$$



$$\text{trace}\ \mathbf{B}\mathbf{A}^\top = \sum_i \mathbf{a}_i^\top \mathbf{b}_i$$