

# OpenGL – Camera συνέχεια

---

Βασιλική Σταμάτη  
Μέλος Ε.ΔΙ.Π. του ΤΜΗΥΠ  
[vstamati@uoi.gr](mailto:vstamati@uoi.gr)  
[vstamati@cse.uoi.gr](mailto:vstamati@cse.uoi.gr)

# MVP matrix

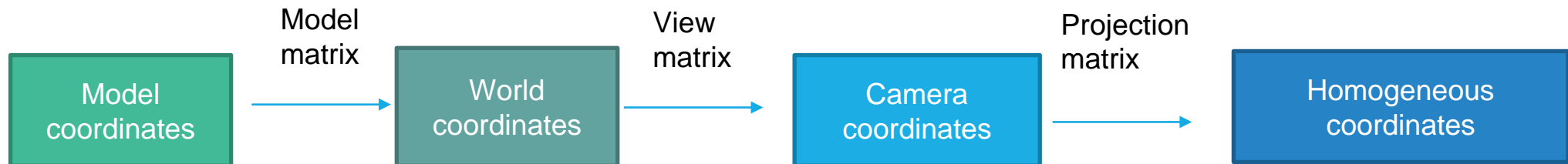
---

Example of transformation matrices: MVP matrix which a matrix that transforms a model from its model space to the screen.

- **Model matrix**
- **View matrix**
- **Projection matrix**

**World space** - All vertices of all models are expressed in reference to the center of the world

**Camera space** – All vertices described in relation to the camera



Για να φτιάξουμε την κάμερα χρειάζεται:

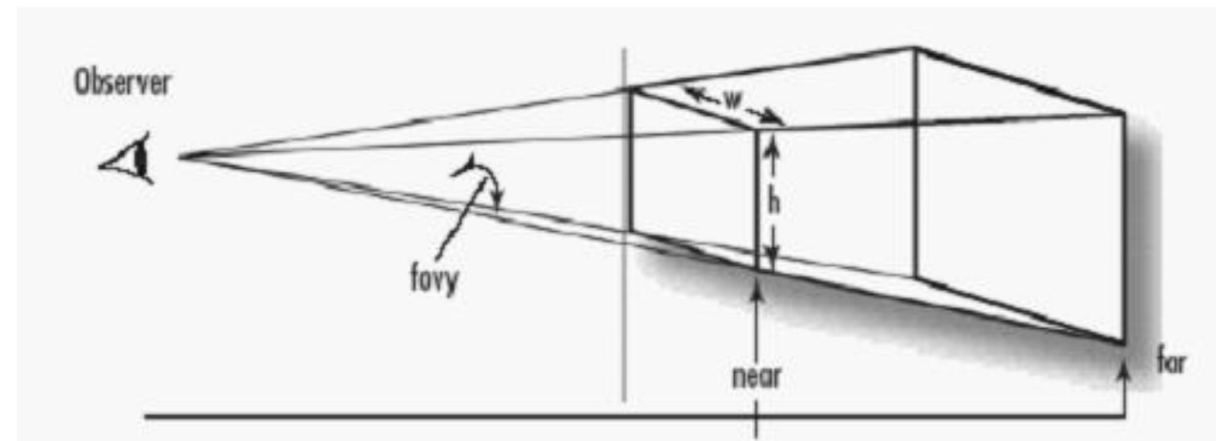
## Projection Matrix

```
detail::tmat4x4<T> glm::perspective ( T const & fovy,  
                                       T const & aspect,  
                                       T const & near,  
                                       T const & far  
                                       )
```

Creates a matrix for a symetric perspective-view frustum.

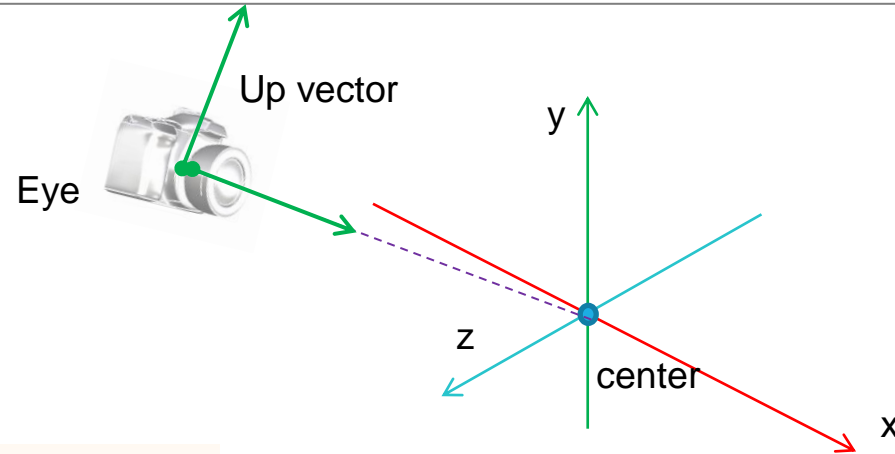
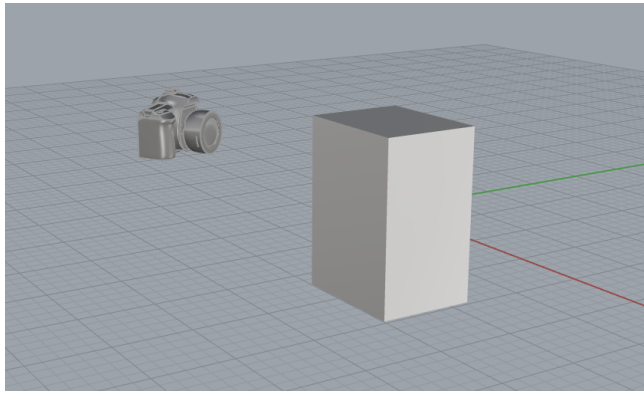
### Parameters

**fovy** Expressed in radians if GLM\_FORCE\_RADIANS is define or degrees otherwise.  
**aspect**  
**near**  
**far**



# Για να φτιάξουμε την κάμερα χρειάζεται:

## View Matrix



```
GLM_FUNC_DECL detail::tmat4x4<T, P> glm::lookAt ( detail::tvec3< T, P > const & eye,  
                                                    detail::tvec3< T, P > const & center,  
                                                    detail::tvec3< T, P > const & up  
                                                    )
```

Build a look at view matrix.

### Parameters

- eye** Position of the camera
- center** Position where the camera is looking at
- up** Normalized up vector, how the camera is oriented. Typically (0, 0, 1)

- Από το up vector και το διάνυσμα κατεύθυνσης μπορούμε να βρούμε με εξωτερικό γινόμενο το διάνυσμα της 3<sup>ης</sup> διάστασης (το κάθετο διάνυσμα στα άλλα δύο – κανόνας τριών δακτύλων)
- Το up vector μας δίνει τον προσανατολισμό της κάμερας στο χώρο

# Μία κάμερα στην OpenGL:

---

- Μπορεί να υλοποιηθεί με διάφορους τρόπους ανάλογα με το ζητούμενο: **κάθε φορά υπολογίζουμε τα ορίσματα της lookat ανάλογα με το αποτέλεσμα που θέλουμε**
  - eye -> υπολογίζουμε τις νέες συντεταγμένες της κάμερας ανάλογα με τον τρόπο που θέλουμε να κινείται  
π.χ. θέλουμε να κινείται γύρω από συγκεκριμένο άξονα, να κινείται πάνω σε συγκεκριμένη τροχιά, κατά μήκος μιας ευθείας κτλ
  - center -> π.χ. κοιτάμε πάντα «μπροστά» (αλλάζει το σημείο που κοιτάει η κάμερα) ή κοιτάμε συνεχώς ένα σταθερό σημείο
  - up -> πώς είναι τοποθετημένη η κάμερα στο χώρο? = προσανατολισμός – δεν ξεχνάμε ότι η κάμερα έχει δικό της τοπικό σύστημα συντεταγμένων που δεν συμπίπτει υποχρεωτικά με το παγκόσμιο σύστημα συντεταγμένων

# Μία κάμερα στην OpenGL:

---

- Μπορεί να ελεγχθεί είτε με το ποντίκι, είτε με το πληκτρολόγιο, είτε με συνδυασμό των δύο
- Στο παράδειγμα που σας έχει δοθεί από το opengl-tutorial (<http://www.opengl-tutorial.org/beginners-tutorials/tutorial-6-keyboard-and-mouse/>) η κάμερα ελέγχεται από το ποντίκι και δεν «βλέπει» συνεχώς το αντικείμενο αλλά πάντα κοιτάει «μπροστά» - προσομοιώνει first person camera
- `deltatime` -> το χρησιμοποιούμε για να μην είναι machine-dependent η υλοποίησή μας – ταχύτητα εκτέλεσης διαφέρει από μηχάνημα σε μηχάνημα

# References

---

Camera tutorials and helpful material:

<http://www.opengl-tutorial.org/>

<https://learnopengl.com/Getting-started/Camera>

[http://www.songho.ca/opengl/gl\\_camera.html](http://www.songho.ca/opengl/gl_camera.html)