# From Face Landmark to Avatar

The goal in this project is to learn a mapping from images of facial expressions into a given 3D avatar. Using deep learning there are descent free tools that can extract 3D facial landmarks and global pose transformation of a person’s head given video frames.

In parallel, the head of a 3D avatar is controlled by a set of rigging points whose movements deform the face of the avatar in a natural manner. In this project you will get a large set of landmarks associated with various facial expressions and their corresponding rigging parameterization. Your goal is to find a transformation between the landmarks and the rigging coordinates. Finally, you will modify this mapping into a dynamic scene using temporal coherence.

Your tasks in this project:

1. Generate data:
   1. Using unity we generate set of avatar frames.
   2. Load the frames into OpenFace.
   3. For each individual image OpenFace will create a corresponding CSV file in the processed directory.

Each file includes:

* + 1. Face
    2. Confidence
    3. Pose\_Tx / pose\_Ty / pose\_Tz
    4. Pose\_Rx / pose\_Ry / pose\_Rz
    5. X0 – X67
    6. Y0 – Y67
    7. Z0 – Z67
  1. To preprocess these CSV file we use the supplied python jupyter notebook.
  2. The preprocess CSV file includes:
     1. X0 – X67
     2. Y0 – Y67
     3. Z0 – Z67
     4. XL0 / YL0 / ZL0 – XL67 / YL67 / ZL67
     5. Quat\_x
     6. Quat\_y
     7. Quat\_z
     8. Quat\_w

1. Train a DNN to apply a mapping from a set of facial landmarks + pose into avatar rigging coordinates.
   1. Input: preprocess CSV files (1.e).
   2. Output: expression file that includes:
      1. Quat\_x
      2. Quat\_y
      3. Quat\_z
      4. Quat\_w
      5. Blendshape\_0 – Blendshape\_67
2. Implement your mapping on a series of facial points and provide visual (qualitative) and quantitative comparisons between the source and the target faces.
3. Improve your mappings using temporal coherence.
4. Provide code + final report.

משימות:

* יש להבין לעומק את התוצרים של Unity, OpenFace וה-Jupiter Notebook: - **מור וגל** **עד 10.08**
  + X, Y , Z של כל פיצ'ר שנוצר על ידי OpenFace
  + Pose של X, Y, Z שנוצר על ידי OpenFace
  + XL, YL, ZL של כל פיצ'ר שנוצר באמצעות מחברת ה-Jupiter
  + Quaternion
  + Blend shape
* להכין data (סט של 50,000 ל-train + 10,000 ל-test) - **מור וגל** **עד 11.08**
  + יצירת ה-frames באמצעות unity – קובץ csv אחוד לכל ה-frames
  + זיהוי הפיצ'רים באמצעות OpenFace ויצירת קבצי ה-csv – קובץ בודד לכל frame
  + ביצוע preprocess לקבצי ה-csv באמצעות מחברת ה-Jupiter notebook - קובץ בודד לכל frame
  + איחוד הקבצים הבודדים של ה-frames (בהתאמה לתוצר של Unity) לקובץ json אחוד

תוצאה: קבלת 60,000 קבצי csv מוכנים להרצה של הרשת נוירונים (קבצים אחודים)

* לבנות רשת נוירונים שמקבלת 412 פיצ'רים (1.e) ומוציאה 67 פיצ'רים + 4 Quaternion על סט 50,000 דוגמאות אימון + לאחר מכן לבצע test על 10,000 הדוגמאות test. – **גל ומור** **ללמוד CNN עד 13.08**
  + יש להכין קובץ csv אחד שמכיל את ה-input וקובץ נוסף שמכיל את ה-output (שקיבלנו באמצעות unity)
  + להחליט על סוג הרשת – DNN פשוט או CNN? (לדעתי צריך ללכת לכיוון CNN ולמצוא דוגמא למימוש כזה באינטרנט)
  + לבנות רשת שכזו:
    - מספר שכבות
    - Activation function
    - Filters
    - Loss function (יש להתייעץ עם טוקי)
  + לאמן אותה באמצעות 50,000 דוגמאות ה-train
  + לבחון את הדיוק שלנו על 10,000 דוגמאות ה-test (יש לקבל loss function מתאים מטוקי)
* אחרי שהגענו לדיוק מספק, לקחת clip שמכיל עשרה frames ולבחון את למידת הרשת + וויזואליזציה