Eye-tracking Literature Review

Na Zhang

Tools

- [2016] Repository for Eye Gaze Detection and Tracking –
- [2019] Webcam-based eye tracking system –Python2/3
  - This is a Python (2 and 3) library that provides a webcam-based eye tracking system.
  - It gives you the exact position of the pupils and the gaze direction, in real time.

❑ [2021] Appearance-based Gaze Estimation With Deep Learning_A Review and Benchmark

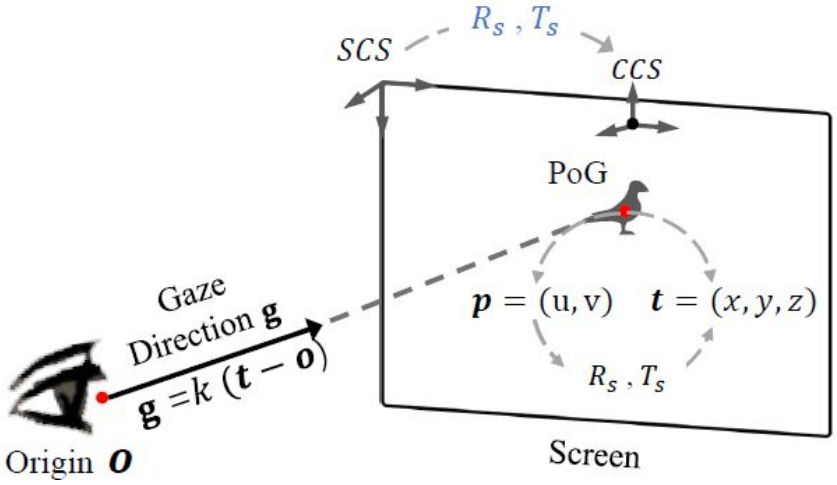
- Website, Paper
- This work helps you quickly build up your own gaze estimation system with standard tools and data, and also compare it with different existing methods.
- In this page, you can find:
  - **Codes** of state-of-the-art gaze estimation methods.
  - **Processed datasets** of gaze estimation.
  - **Tools** for the practice of gaze estimation.

Benchmark

- Our survey builds a comprehensive benchmark.
- Please refer to our paper for more details.
- The benchmark contains various kinds of gaze estimation methods.
- Note that different methods have different outputs (2D gaze positions or 3D gaze directions).
- Therefore, we provide:
  - codes for post-processing (2D->3D and 3D->2D) to enable fair comparisons among different methods.
  - publish standard tools for data rectification and gaze origin conversion.

Code 1: 2D/3D Gaze Conversion

- 3D gaze means gaze directions originating from subjects and pointing to gaze targets.
- 2D gaze means the **point of gaze (PoG)**. It is also the coordinate of gaze targets in the screen. We illustrate them in the figure below.
- Please download the core code here.



- **3D Gaze To 2D Gaze**
  - We provide an example.
  - `gazepoint = Gaze3DTo2D(gazedirection, gazeorigin, rmat, tmat, require3d)`

  - We also provide specific conversion codes in each dataset.
    - MPIIGaze
    - EyeDiap
  - To use the specific codes, you first should download the core code and rename the code as "`conversion.py`". The document about the specific codes is coming soon.

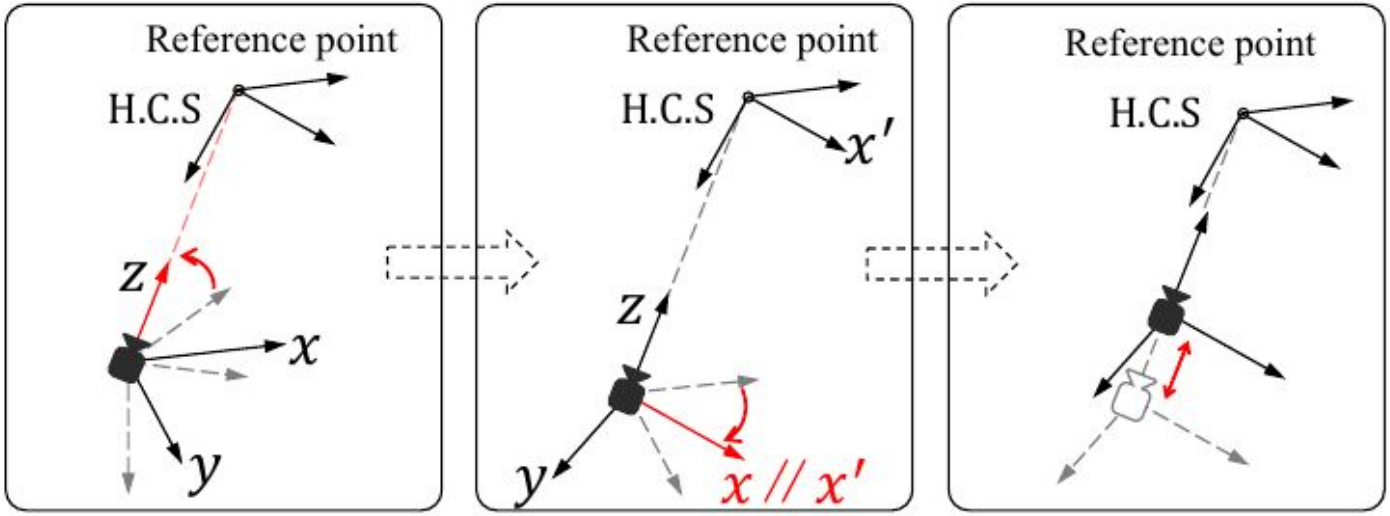
- **2D Gaze To 3D Gaze**
  - We also provide an example.
  - `gazepoint = Gaze2DTo3D(point, origin, rmat, tmat)`

  - We also provide specific conversion codes in each dataset.
    - MPIIGaze
    - EyeDiap
  - To use the specific codes, you first should download the core code and rename the code as "`conversion.py`". The document about the specific codes is coming soon.

Code 2: Data rectification

- The goal of data rectification is to eliminate the environmental factors by data pre-processing methods and to simplify the gaze regression problem.
- Current data rectification methods mainly focus on the factors of the head pose and the illumination.
- We follow the process of [1].
- We assume that the captured eye image is a plane in 3D space, the rotation of the virtual camera can be performed as a perspective transformation on the image.
- The whole data rectification process is shown in the figure below.

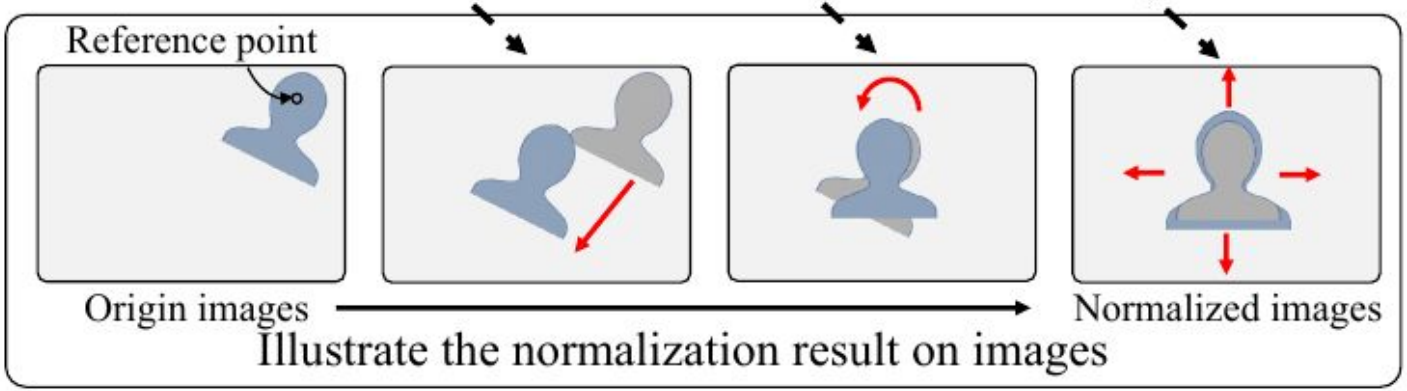
[1] MPIIGaze: Real-World Dataset and Deep Appearance-Based Gaze Estimation



(a). The z-axis points out the reference point

(b). The x-axis parallels to x'-axis of H.C.S.

(c). Keep the same distance to the reference point



Illustrate the normalization result on images

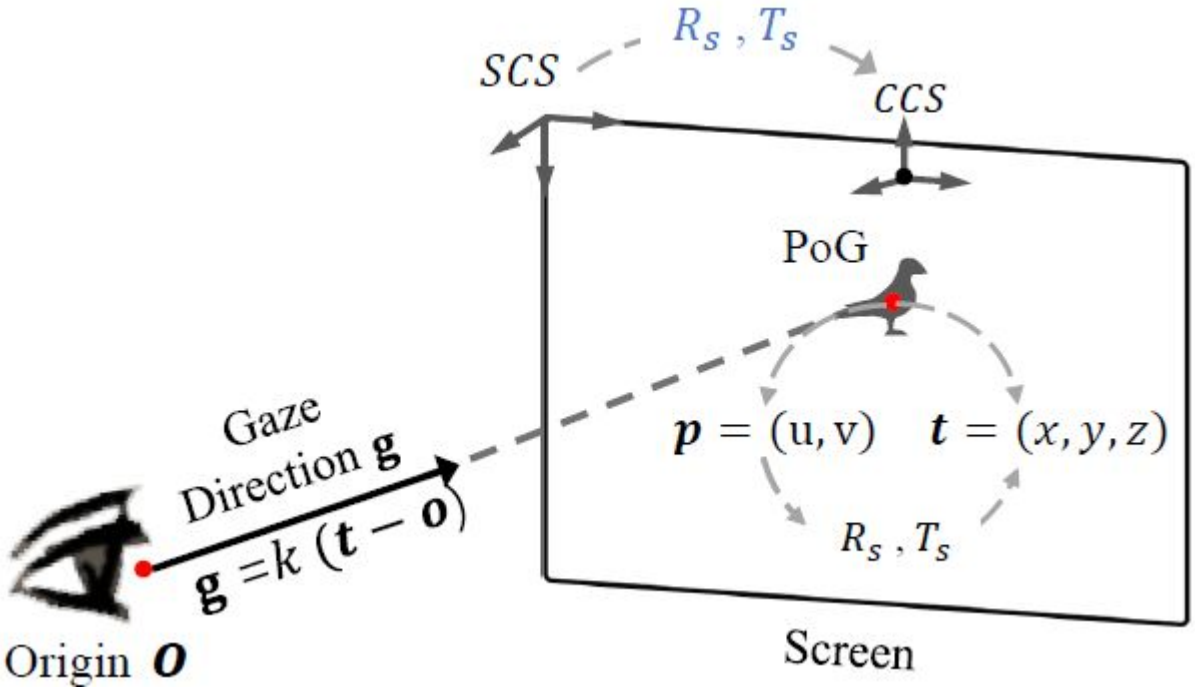
- We first define a reference point which is usually eye center or face center.
- Then, we rotate the virtual camera where the rotated camera is pointing towards the reference point.
- This operation cancel the variance caused by camera position.
- We also rotate the virtual camera so that the appearance captured by the rotated cameras is facing the front.
- Finally, we scale the image to ensure the same distance between subjects and cameras.
- Note that, the above operation only explain the rectification in image aspect.
- The gaze direction also should transformed.
- Please refer our paper for more details.

- **Illumination** also influences the appearance of the human eye.
- To handle this, researchers usually take gray-scale eye images rather than RGB eye images as input and apply histogram equalization in the gray-scale images to enhance the image.

- Please download the code `data_processing_core.py` here.
- We also provide an example of normalizing one image.

Code 3: Gaze Origin Conversion

- Some methods estimate the gaze direction originating from `eye center`, and some methods estimate the gaze direction originating from `face center`.
- To convert the two kinds of gaze direction, we need to first compute the corrdinate of PoR t in CCS.
- Then, we acquire the gaze origin o and further compute the new gaze direction.


R_s , T_s
SCS
CCS
PoG
Gaze
Direction $\mathbf{g}$
$\mathbf{g} = k\,(\mathbf{t} - \mathbf{o})$
$\boldsymbol{p} = (\mathrm{u}, \mathrm{v})$
$\boldsymbol{t} = (x, y, z)$
R_s , T_s
Origin $\boldsymbol{o}$
Screen

- Please download the core code here.
- We also provide an example.
- `gaze = ToNewOrigin(gaze, origin, neworigin, rmat, tvec)`

- We also provide specific conversion codes in each dataset.
  - MPIIGaze
  - EyeDiap
- To use the specific codes, you first should download the core code and rename the code as "`conversion.py`". The document about the specific codes is coming soon.

Benchmark of 3D Gaze Estimation

Methods \ Datasets	Task A: Estimate gaze directions originating from eye center			Task B: Estimate gaze directions originating from face center			
	MPIIGaze [1]	EyeDiap [2]	UT [3]	MPIIFaceGaze [9]	EyeDiap [2]	Gaze360 [4]	ETH-XGaze [5]
Proposed for task A	**Direct results of task A**			**Converted results from task A**			
Mnist [6]	6.27°	7.60°	6.34°	6.39°	7.37°	N/A	N/A
GazeNet [1]	5.70°	7.13°	6.44°	5.76°	6.79°	N/A	N/A
Proposed for task B	**Converted results from task B**			**Direct results of task B**			
Dilated-Net [7]	4.39°	6.57°	N/A	4.42°	6.19°	13.73°	N/A
Gaze360 [4]	4.07°	5.58°	N/A	4.06°	5.36°	11.04°	4.46°
RT-Gene [8]	4.61°	6.30°	N/A	4.66°	6.02°	12.26°	N/A
FullFace [9]	4.96°	6.76°	N/A	4.93°	6.53°	14.99°	7.38°
CA-Net [10]	4.27°	5.63°	N/A	4.27°	5.27°	11.20°	N/A
Proposed for task C (In Tab. VI)	**Converted results from task C** (In Tab. VI)			**Converted results from task C** (In Tab. VI)			
Itracker [11]	7.25°	7.50°	N/A	7.33°	7.13°	N/A	N/A
AFF-Net [12]	3.69°	6.75°	N/A	3.73°	6.41°	N/A	N/A

Benchmark of 2D Gaze Estimation

Methods \ Datasets	Task C: Estimate 2D gaze points.			
	MPIIFaceGaze [9]	EyeDiap [2]	GazeCapture [11]	
			Tablet	Phone
Proposed for task C	**Direct results of task C**			
Itracker [11]	7.67 cm	10.13 cm	2.81 cm	1.86 cm
AFF-Net [12]	4.21 cm	9.25 cm	2.30 cm	1.62 cm
SAGE [13]	N/A	N/A	2.72 cm	1.78 cm
TAT [14]	N/A	N/A	2.66 cm	1.77 cm
Proposed for task A	**Converted results from task A** (In Tab. V)			
Mnist [6]	7.29 cm	9.06 cm	N/A	N/A
GazeNet [1]	6.62 cm	8.51 cm	N/A	N/A
Proposed for task B	**Converted results from task B** (In Tab. V)			
Dilated-Net [7]	5.07 cm	7.36 cm	N/A	N/A
Gaze360 [4]	4.66 cm	6.37 cm	N/A	N/A
RT-Gene [8]	5.36 cm	7.19 cm	N/A	N/A
FullFace [9]	5.65 cm	7.70 cm	N/A	N/A
CA-Net [10]	4.90 cm	6.30 cm	N/A	N/A

Reference

1. **MPIIGaze**: Real-World Dataset and Deep Appearance-Based Gaze Estimation
2. **EYEDIAP** Database: Data Description and Gaze Tracking Evaluation Benchmarks
3. Learning-by-Synthesis for Appearance-based 3D Gaze Estimation
4. **Gaze360**: Physically Unconstrained Gaze Estimation in the Wild
5. **ETH-XGaze**: A Large Scale Dataset for Gaze Estimation under Extreme Head Pose and Gaze Variation
6. Appearance-Based Gaze Estimation in the Wild
7. Appearance-Based Gaze Estimation Using Dilated-Convolutions
8. RT-GENE: Real-Time Eye Gaze Estimation in Natural Environments
9. It’s written all over your face: Full-face appearance-based gaze estimation
10. A Coarse-to-fine Adaptive Network for Appearance-based Gaze Estimation
11. Eye Tracking for Everyone
12. Adaptive Feature Fusion Network for Gaze Tracking in Mobile Tablets
13. On-Device Few-Shot Personalization for Real-Time Gaze Estimation
14. A Generalized and Robust Method Towards Practical Gaze Estimation on Smart Phone

Content

- Over the last decades, a plethora of gaze estimation methods has been proposed.
- These methods usually fall into three categories:
  - 3D eye model recovery-based method
  - 2D eye feature regression-based method
  - Appearance-based method.
- 3D eye model recovery-based methods construct a geometric 3D eye model and estimates gaze directions based on the model.
- The 3D eye model is usually person-specific due to the diversity of human eyes.
- Therefore, these methods usually require personal calibration to recover person-specific parameters such as iris radius and kappa angle.
- The 3D eye model recovery-based methods usually achieve reasonable accuracy while they require dedicated devices such as infrared cameras.

- The 2D eye feature regression-based methods usually keep the same requirement on devices as 3D eye model recovery-based methods.
- The methods directly use the detected geometric eye feature such as pupil center and glint to regress the **point of gaze (PoG)**.
- They do not require geometric calibration for converting gaze directions into PoG.

- Appearance-based methods do not require dedicated devices, instead, it uses on-the-shelf web cameras to capture human eye appearance and regress gaze from the appearance.
- Although the setup is simple, it usually requires the following components:
  - 1) **An effective feature extractor** to extract gaze features from high-dimensional raw image data.
    - Some feature extractors such as histograms of oriented gradients are used in the conventional method [14].
    - However, it can not effectively extract high-level gaze features from images.
  - 2) **A robust regression function** to learn the mappings from appearance to human gaze.
    - It is non-trivial to map the high-dimensional eye appearance to the low-dimensional gaze.
    - Many regression functions have been used to regress gaze from appearance, e.g., local linear interpolation [15], adaptive linear regression [13] and gaussian process regression [16] , the regression performance is barely satisfactory.
  - 3) **A large number of training samples** to learn the regression function.
    - They usually collects personal samples with a time-consuming personal calibration,and learns a person-specific gaze estimation model.
    - Some studies seek to reduce the number of training samples. Lu et al. propose an adaptive linear regression method to select an optimal set of sparsest training sample for interpolation [13].
    - However, this also limits the usage in real-world applications.

DL appearance-based gaze estimation methods review

- In this paper, we provide a comprehensive review of appearance-based gaze estimation methods in deep learning.
- As shown in Fig. 1, we discuss these methods from four perspectives:
  - 1) deep feature extraction,
  - 2) deep neural network architecture design,
  - 3) personal calibration,
  - 4) device and platform.

- deep feature extraction
  - how to extract effective feature
  - Three raw feature:
    - eye images
    - face images
    - videos.

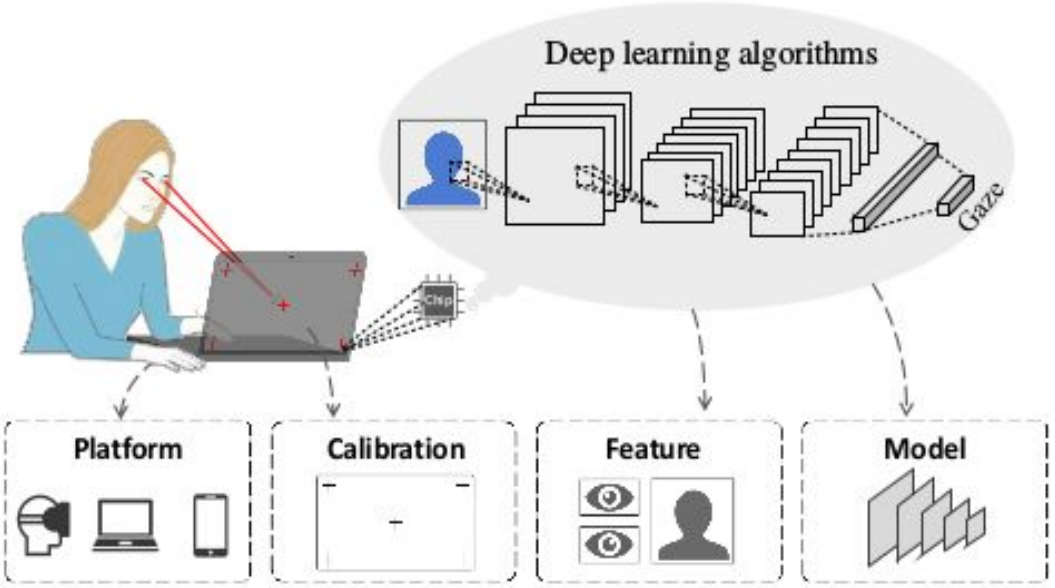


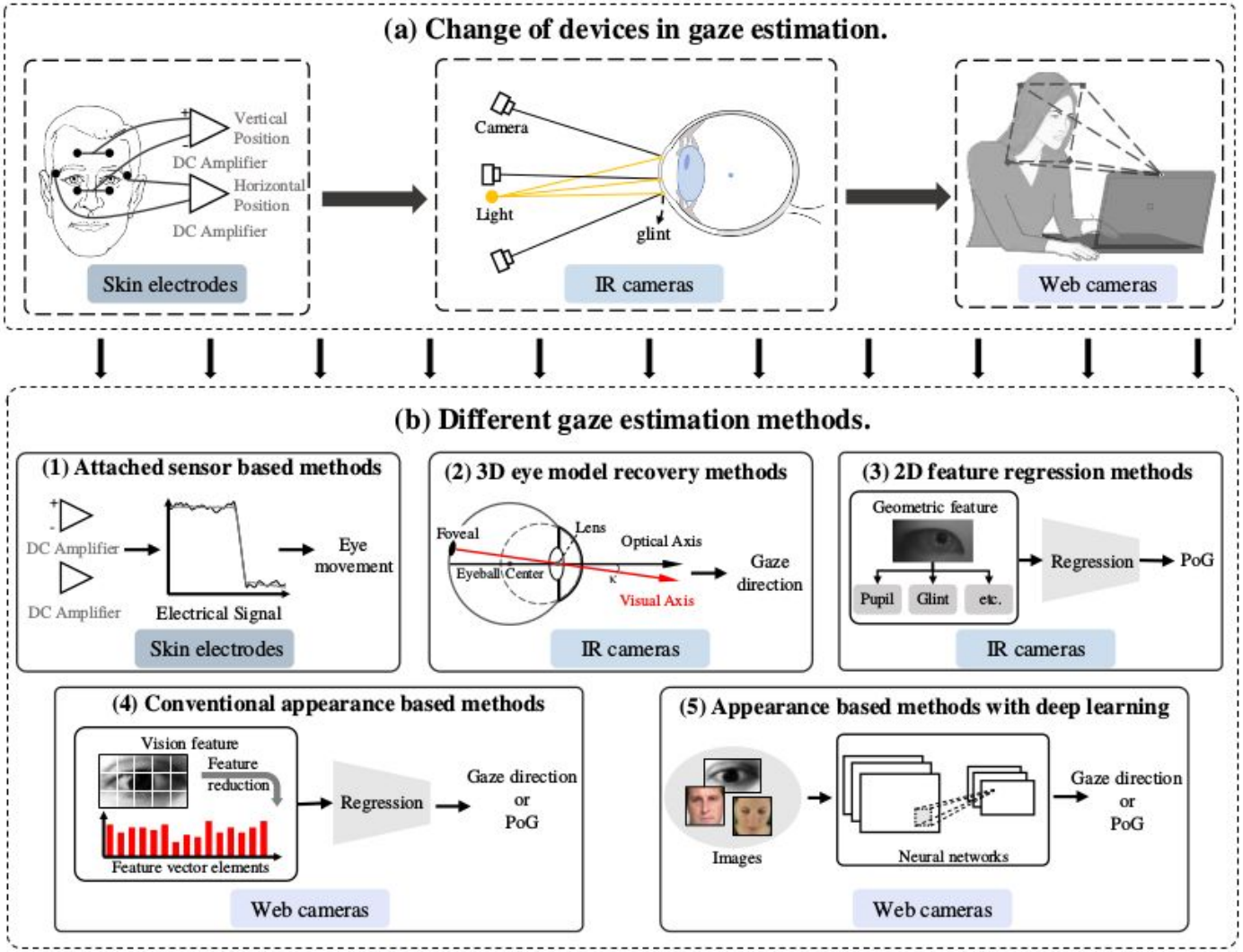
Fig. 1. Deep learning based gaze estimation relies on simple devices and complex deep learning algorithms to estimate human gaze. It usually uses off-the-shelf cameras to capture facial appearance, and employs deep learning algorithms to regress gaze from the appearance. According to this pipeline, we survey current deep learning based gaze estimation methods from four perspectives: deep feature extraction, deep neural network architecture design, personal calibration as well as device and platform.

- deep neural network architecture design:
  - review advanced CNN models
  - According to the supervision mechanism, we respective review supervised, self-supervised, semi-supervised and unsupervised gaze estimation
- personal calibration
  - How to use calibration samples to further improve the performance of CNNs.
  - introduce the method integrating userunaware calibration sample collection mechanism.
- device and platforms
  - different cameras, i.e., RGB cameras, IR cameras and depth cameras
  - different platforms, i.e., computer, mobile devices and head-mount device.

- the practice of gaze estimation
  - data pre-processing methods:
    - face and eye detection methods
    - common data rectification methods.
  - various forms of human gaze, e.g.,
    - gaze direction
    - PoG
  - data post-processing methods
    - geometric conversion between various human gaze.
- build gaze estimation benchmarks based on the data post-processing methods.

Gaze Estimation Background

- Figure 2 illustrates the development progress of gaze estimation methods.
- **Early gaze estimation methods** rely on detecting eye movement patterns such as fixation, saccade and smooth pursuit [11].
- They attach the sensors around the eye and use potential differences to measure eye movement [17], [18].
- With the development of computer vision technology, **modern eye-tracking devices** have emerged. These methods usually estimate gaze using the eye/face images captured by a camera.
- In general, there are two types of such devices, the remote eye tracker and the head-mounted eye tracker.
- The remote eye tracker usually keeps a certain distance from the user, typically ~60 cm.
- The head-mounted eye tracker usually mounts the cameras on a frame of glasses.
- Compared to the intrusive eye tracking devices, the modern eye tracker greatly enlarges the range of application with computer vision-based methods.


(a) Change of devices in gaze estimation.
Vertical Position
DC Amplifier
Horizontal Position
DC Amplifier
Skin electrodes
Camera
Light
glint
IR cameras
Web cameras
(b) Different gaze estimation methods.
(1) Attached sensor based methods
DC Amplifier
DC Amplifier
Electrical Signal
Eye movement
Skin electrodes
(2) 3D eye model recovery methods
Fovea
Lens
Optical Axis
Eyeball Center
Visual Axis
Gaze direction
IR cameras
(3) 2D feature regression methods
Geometric feature
Pupil
Glint
etc.
Regression
PoG
IR cameras
(4) Conventional appearance based methods
Vision feature
Feature reduction
Feature vector elements
Regression
Gaze direction or PoG
Web cameras
(5) Appearance based methods with deep learning
Images
Neural networks
Gaze direction or PoG
Web cameras

- **Computer vision-based methods** can be further divided into three types:
  - the 2D eye feature regression method
  - the 3D eye model recovery method
  - the appearance-based method.
- The first two types of methods estimate gaze **based on detecting geometric features** such as contours, reflection and eye corners.
- The geometric features can be accurately extracted with the assistance of **dedicated devices, e.g., infrared cameras**.
- Detailly, the 2D eye feature regression method learns a mapping function from the geometric feature to the human gaze, e.g., the polynomials [19], [20] and the neural networks [21].
- The 3D eye model recovery method builds a subject-specific geometric eye model to estimate the human gaze.
- The eye model is fitted with geometric features, such as the infrared corneal reflections [22], [23], pupil center [24] and iris contours [25].
- In addition, the eye model contains subject-specific parameters such as cornea radius, kappa angles.
- Therefore, it usually **requires time-consuming personal calibration** to estimate these subject-specific parameters for each subject.

- **Appearance-based methods:**
  - directly learn a mapping function from images to human gaze.
  - Different from 2D eye feature regression methods, appearance-based methods **do not require dedicated devices for detecting geometric features**.
  - They **use image features** such as image pixel [13] or deep features [26] **to regress gaze**.
  - Various regression models have been used, e.g., :
    - the neural network [27],
    - the Gaussian process regression model [16],
    - the adaptive linear regression model [13]
    - the convolutional neural network [26].
    - Transformer ??
  - However, this is still a challenging task due to the complex eye appearance.

Appearance-based Gaze Estimation

- **directly learn the mapping function from eye appearance to human gaze.**
- Early appearance-based methods usually learn a subject-specific mapping function.
  - require a time-consuming calibration to collect the training samples of the specific subject.
  - However, these methods only show reasonable performance in a constrained environment, i.e., fixed head pose and the specific subject.
  - Their performance significantly degrades when tested on an unconstrained environment.
  - This problem is always challenging in appearance-based gaze estimation.

- To address the performance degradation problem across subjects
  - Funes et al. presented a cross-subject training method [29].
  - Sugano et al. introduce a learning-by-synthesis method [30].
  - Lu et al. employ a sparse auto-encoder to learn a set of bases from eye image patches and reconstruct the eye image using these bases [31].
- To tackle the head motion problem
  - Sugano et al. cluster the training samples with similar head poses and interpolate the gaze in local manifold [32].
  - Lu et al. suggest that initiating the estimation with the original training images and compensating for the bias via regression [33].
  - Lu et al. further propose a novel gaze estimation method that handles the free head motion via eye image synthesis using a single camera [34].

- **Deep Learning for Appearance-based Gaze Estimation**
- Conventional appearance-based methods cannot handle these challenges gracefully due to the weak fitting ability.
- an increasing number of improvements and extensions on CNN-based gaze estimation methods emerged.

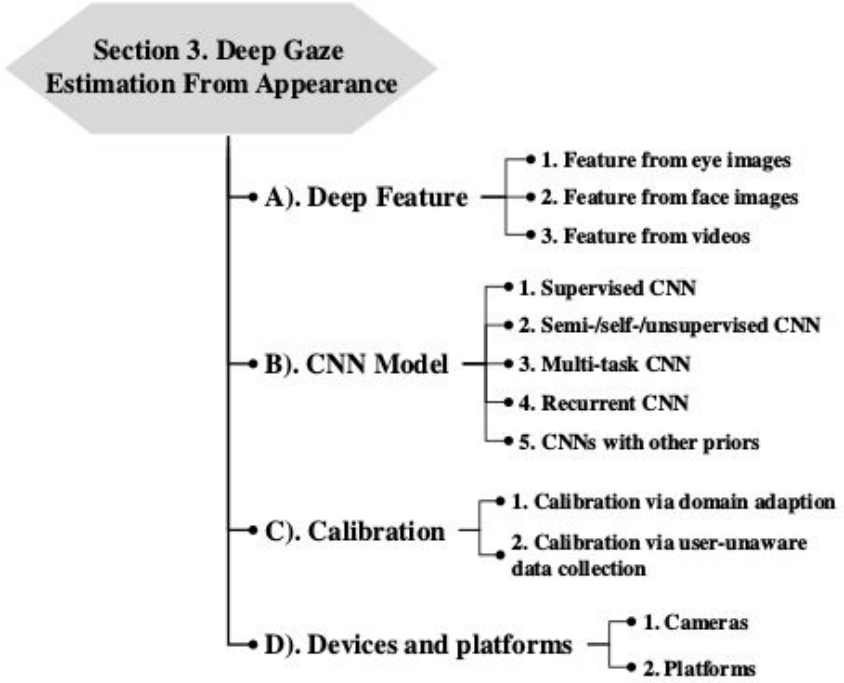


Fig. 3. The architecture of section 3. We introduce gaze estimation with deep learning from four perspectives.

A). Deep Feature From Appearance

- feature extraction method according to the types of input into the deep neural network: **eye images, face images and videos**.
- **1) Feature from eye images:**
  - The gaze direction is highly correlated with the eye appearance.
  - Any perturbation in gaze direction results in eye appearance changes.
  - For example, the rotation of the eyeball changes the location of the iris and the shape of the eyelid, which leads to changes in gaze direction.
  - Early deep learning-based methods estimate the gaze from single eye image.
  - Recent studies found that concatenating the features of two eyes help to improve the gaze estimation accuracy
  - supervised/unsupervised approaches
  - CNN/GAN methods

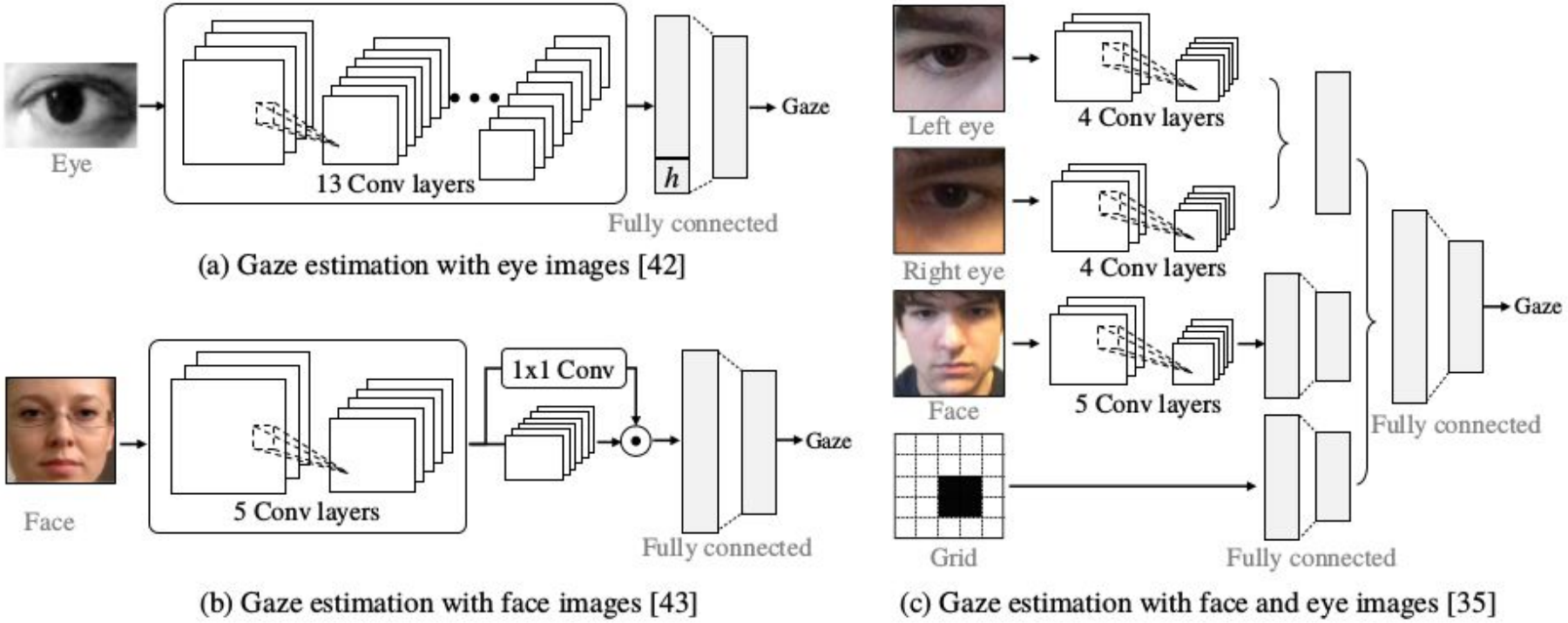


Fig. 4. Some typical CNN based gaze estimation network. (a). Gaze estimation with eye images [42]. (b) Gaze estimation with face images [43]. (c). Gaze estimation with face and eye images [35].

- **2) Feature from face images:**
- Face images contain the head pose information that also contributes to gaze estimation.
- Face images contain redundant information. Researchers have attempted to filter out the useless features in face image [43], [58].
- Some studies crop the eye image out of the face images and directly feed it into the network.
- Facial landmarks have also been used as additional features to model the head pose and eye position.

- **3) Feature from videos**
- temporal information from the videos also contributes to better gaze estimates
- Recurrent Neural Network (RNN) has been widely used in video processing, e.g., long short-term memory (LSTM) **[36], [70]**.
- As shown in Fig. 5, they usually use a CNN to extract the features from the face images at each frame, and then input these features into a RNN.
- The temporal information is automatically captured by the RNN for gaze estimation
- Temporal features such as the **optical flow and eye movement** dynamics have been used to improve gaze estimation
    - The **optical flow** provides the motion information between the frames [71].
    - **Eye movement dynamics**, such as fixation, saccade and smooth pursuits, have also been used to improve gaze estimation accuracy [72][73].

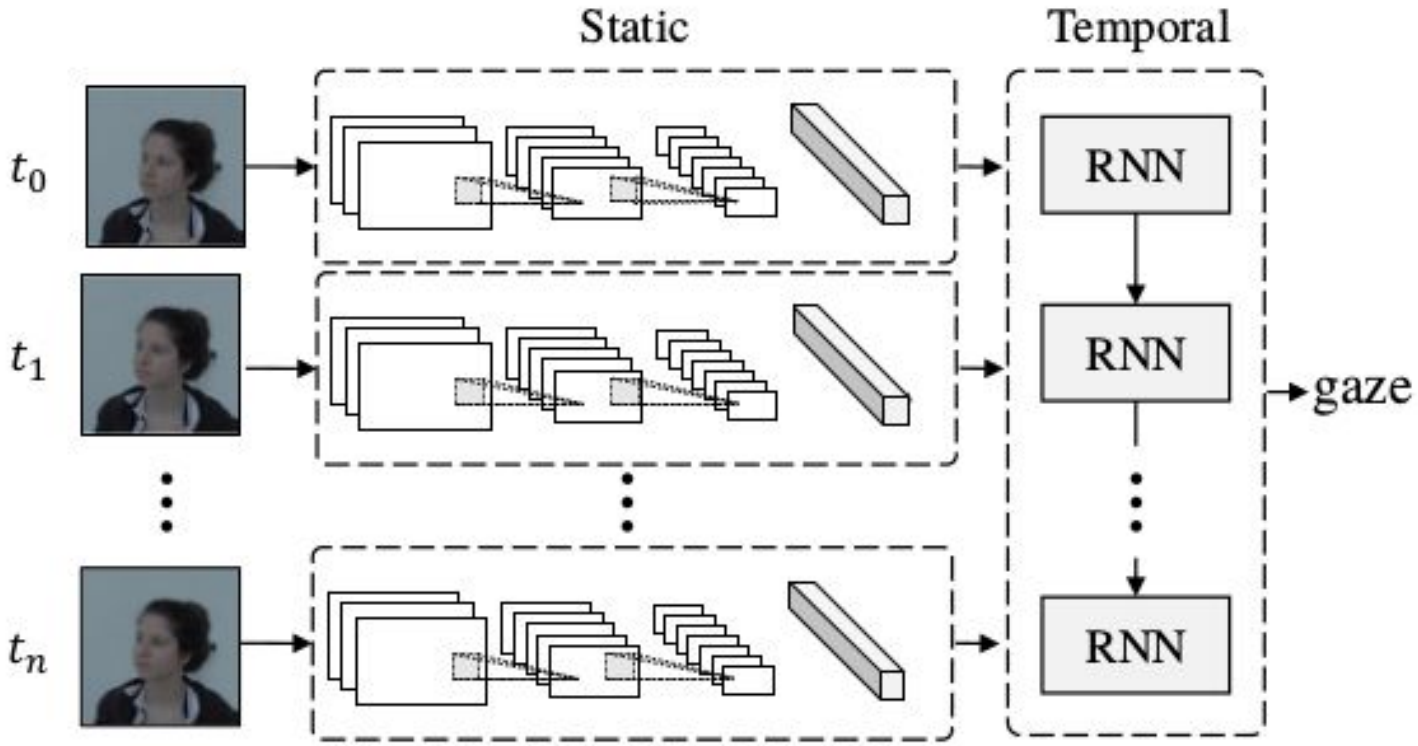


Fig. 5. Gaze estimation with videos. It first extracts static features from each frame using a typical CNN, and feeds these static features into RNN for extracting temporal information.

B). CNN Models

- **Convolutional neural networks**
- 1) Supervised CNNs
  - The network is trained using image samples with ground truth gaze directions.
  - The gaze estimation problem is essentially learning a mapping function from raw images to the human gaze
  - LeNet [26], AlexNet [43], VGG [42], ResNet18 [36] and ResNet50 [82]
  - However, it is difficult and time-consuming to collect enough gaze data in practical applications. Inspired by the physiological eye model [85], some researchers propose to synthesize labeled photo-realistic image [30], [86], [87].
  - These methods usually build eye-region models and render new images from these models.

- 2) Semi-/Self-/Un-supervised CNNs
  - rely more on the unlabeled images to boost the gaze estimation performance.

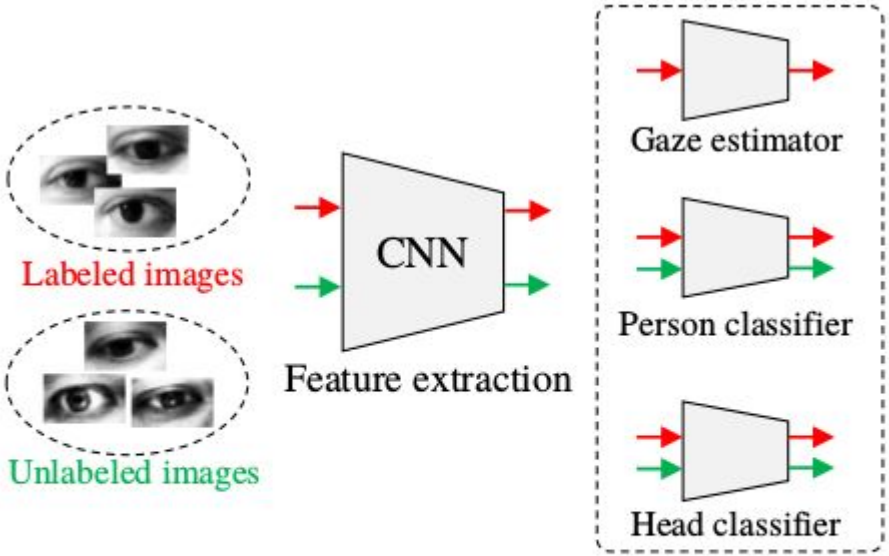


Fig. 6. A semi-supervised CNN [51]. It uses both labeled images and unlabeled images for training. It designs an extra appearance classifier and a head pose classifier. The two classifiers align the feature of labeled images and unlabeled images.

- Self-supervised CNNs aim to formulate a pretext auxiliary learning task to improve the estimation performance.
    - the network contains a regression network to estimate the two eyes' gaze directions, and an evaluation network to assess the reliability of two eyes. During training, the result of the regression network is used to supervise the evaluation network, the accuracy of the evaluation network determines the learning rate in the regression network. They simultaneously train the two networks and improve the regression performance without additional inference parameters.

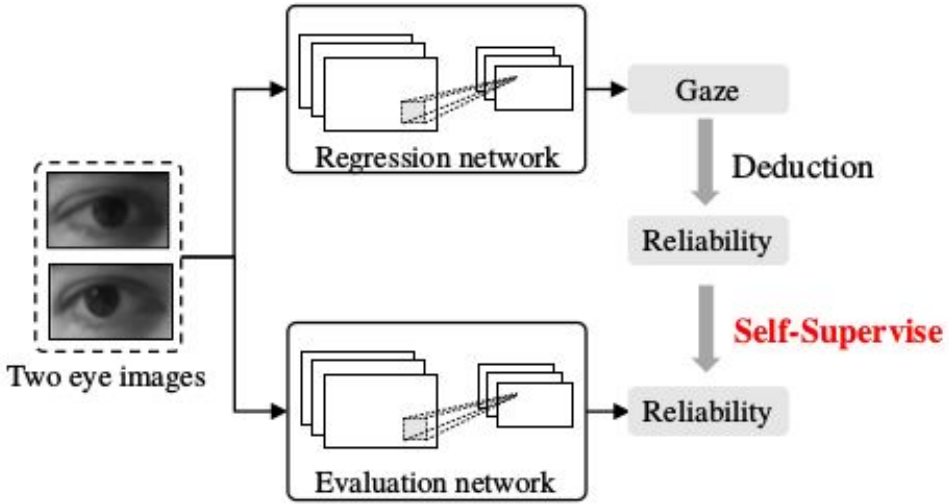


Fig. 7. A self-supervised CNN [48]. The network is consisted of two sub-networks. The regression network estimates gaze from two eye images and generates the ground truth of the other network for self-supervision.

- Unsupervised CNNs only require unlabeled data for training

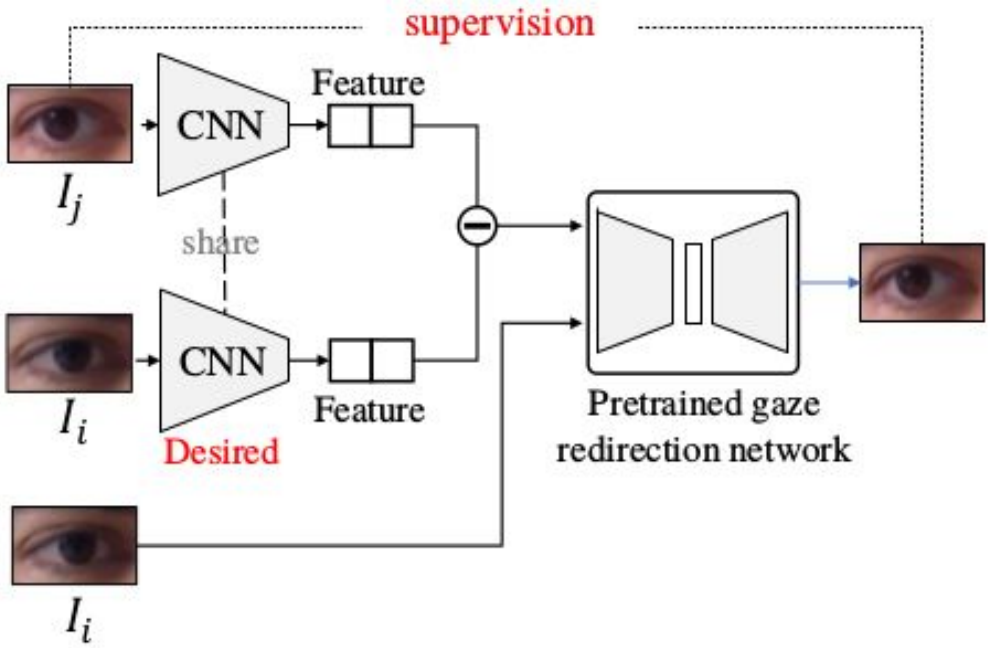


Fig. 8. A unsupervised CNN [54]. It uses a CNN to extract 2-D feature from eye images. The feature difference of two images and one of eye image are fed into a pretrained gaze redirection network to generate the other eye image.

- 3) Multi-task CNNs:
  - Contains multiple tasks that provide related domain information as inductive bias to improve model generalization
  - Estimate gaze directions based on single-view eye images and PoG from multi-view eye images.
  - decompose the gaze into multiple related features and construct multi-task CNNs to estimate these feature.

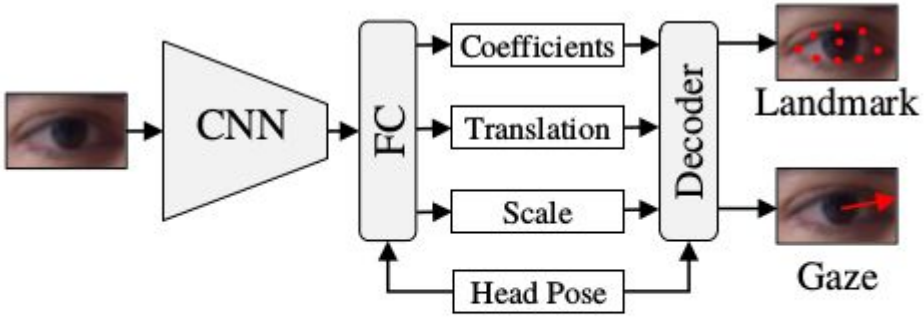


Fig. 9. A multitask CNN [95]. It estimates the coefficients of a landmark-gaze model as well as the scale and translation parameters to align eye landmarks. The three results are used to calculate eye landmarks and estimated gaze.

- 4) Recurrent CNNs:
  - using temporal information.
  - Recently, recurrent neural networks have shown great capability in handling sequential data.
  - employ recurrent CNNs to estimate the gaze in videos[66], [36], [70].

C). Personal Calibration

- Conventional 3D eye model recovery methods usually build a unified gaze model including subject-specific parameters such as eyeball radius [22].
- They perform a personal calibration to estimate these subject-specific parameters.
- In the field of deep learning-based gaze estimation, personal calibration is also explored to improve person-specific performance.

- 1) Calibration via Domain Adaptation:
- 2) Calibration via User-unaware Data Collection:

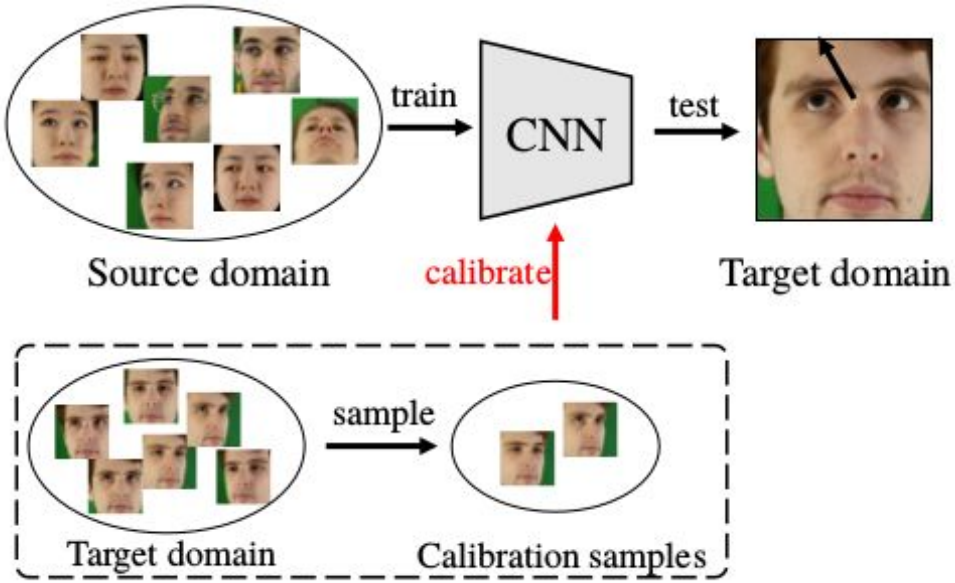


Fig. 10. Personal calibration in deep learning. The method usually samples a few images from the target domain as calibration samples. The calibration samples and training set are jointly used to improve the performance in target domain.

D). Devices and Platforms

- 1) Camera:
  - The majority of gaze estimation systems use a single RGB camera to capture eye images,
  - while some studies use different camera settings, e.g.,
    - using multiple cameras to capture multi-view images [98], [119],
    - using infrared (IR) cameras to handle low illumination condition [100], [121],
    - and using RGBD cameras to provide the depth information [99].

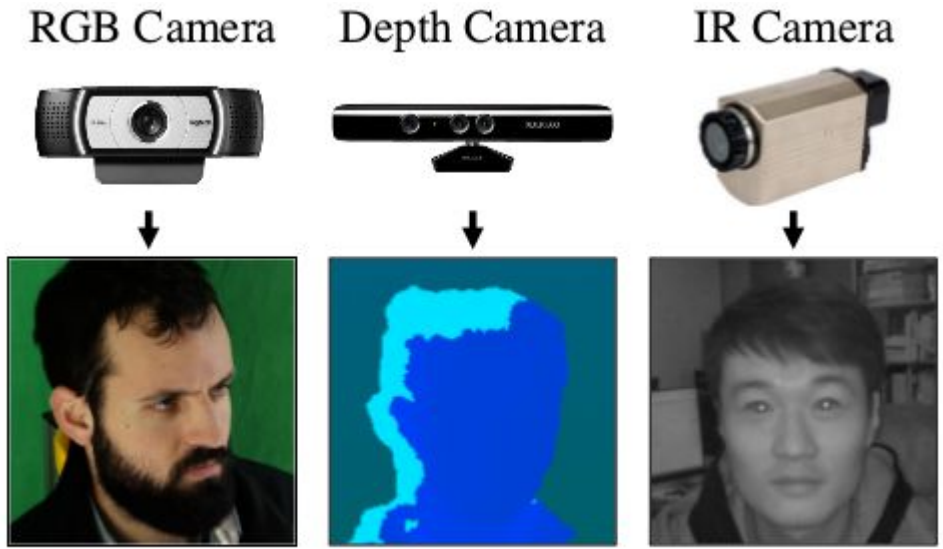


Fig. 11. Different cameras and their captured images.

- 2) Platform:
  - three types of platforms: computers, mobile devices and head-mounted devices.

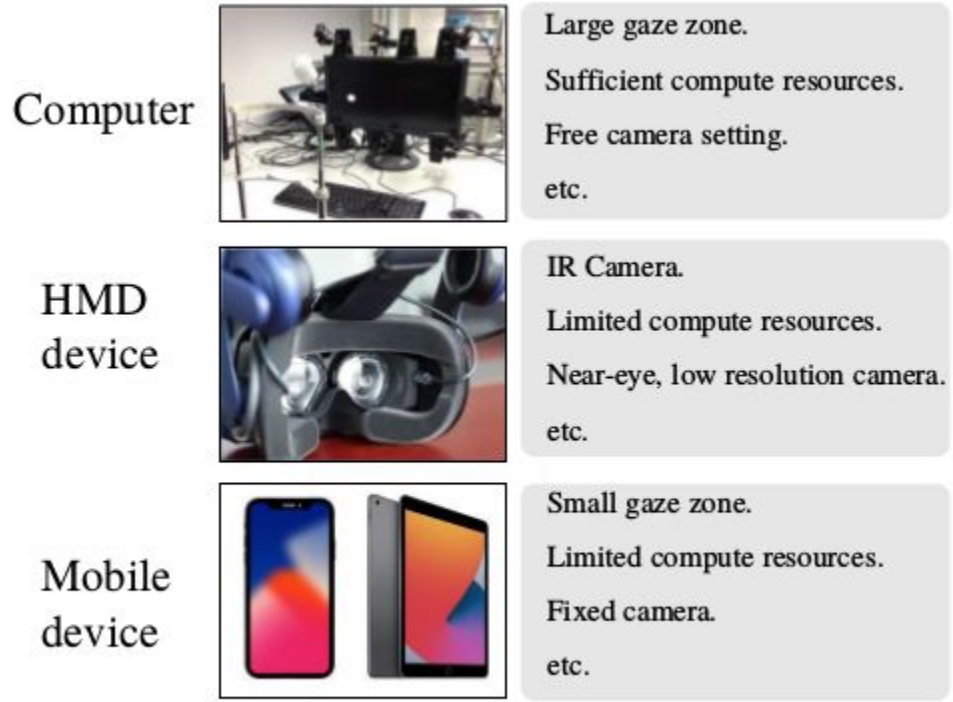


Fig. 12. Different platforms and their characteristics.

TABLE II
SUMMARY OF FACE ALIGNMENT METHODS

Names	Years	Pub.	Links
Dlib [131]	2014	CVPR	https://pypi.org/project/dlib/19.6.0/
MTCNN [132]	2016	SPL	https://github.com/kpzhang93/MTCNN_face_detection_alignment
DAN [133]	2017	CVPRW	https://github.com/MarekKowalski/DeepAlignmentNetwork
OpenFace [134]	2018	FG	https://github.com/TadasBaltrusaitis/OpenFace
PRN [135]	2018	ECCV	https://github.com/YadiraF/PRNet
3DDFA_V2 [136]	2020	ECCV	https://github.com/cleardusk/3DDFA_V2