



**VisionLabs**  
MACHINES CAN SEE

# Configuration Guide

written for LUNA SDK version 5.30.1

## Contents

<b>Configuration Guide</b>	<b>4</b>
Configuration file location . . . . .	4
<b>Settings</b>	<b>6</b>
System settings . . . . .	6
Descriptor factory settings . . . . .	7
FaceDetV3 detector settings . . . . .	9
FaceDetV2 detector settings . . . . .	12
LNet . . . . .	13
LNetIR . . . . .	13
SLNet . . . . .	13
IndexBuilder settings . . . . .	14
HumanDetector settings . . . . .	15
Head detector settings . . . . .	18
Quality estimator settings . . . . .	19
HeadPoseEstimator settings . . . . .	20
AttributeEstimator settings . . . . .	21
EyeEstimator settings . . . . .	22
GlassesEstimator settings . . . . .	23
OverlapEstimator settings . . . . .	24
LivenessFPREstimator settings . . . . .	25
LivenessIREstimator settings . . . . .	26
NIRLivenessEstimator settings . . . . .	27
Mouth Estimator settings . . . . .	28
Face Occlusion Estimator Settings . . . . .	29
DeepFake Estimator settings . . . . .	31
Medical mask estimator settings . . . . .	33
RedEyeEstimator settings . . . . .	35
Depth Estimator settings . . . . .	36
LivenessFlyingFaces Estimator settings . . . . .	37
LivenessRGBM Estimator settings . . . . .	38
LivenessOneShotRGBEstimator settings . . . . .	39
Credibility Estimator settings . . . . .	42
Natural Light Estimator settings . . . . .	43
BlackWhite Estimator settings . . . . .	44
Fish Eye Estimator settings . . . . .	45
Background Estimator settings . . . . .	46
Human Attribute Estimator settings . . . . .	47

Portrait Style Estimator settings . . . . .	50
HumanFace detector settings . . . . .	51
Landmarks detector settings . . . . .	53
Orientation Estimator settings . . . . .	54
ImageModification Estimator settings . . . . .	55
Crowd estimator settings . . . . .	56
LivenessDepthRGBEstimator settings . . . . .	57
DepthLivenessEstimator settings . . . . .	58
FightsEstimator settings . . . . .	59
<b>Runtime settings</b>	<b>60</b>

## Configuration Guide

Configuration options are specified via `faceengine.conf` file which is basically an XML document with special tag formatting. The document itself is not required to exist, in this case FSDK will fall back to some default settings, which, however, may not be suitable for several tasks.

**WARNING!** By changing any configuration settings from default ones it is assumed that user understands what these settings do and how they will affect performance and output results of their application. The rule of thumb is this: DO NOT change anything in configuration file unless you really have to.

Always remember that incorrect config may huck the things up very badly. Pay attention to what you configure and how. Always double-check what you deploy.

Some configuration settings may be omitted due to their obscurity and research use case only.

The location where the config file is found varies across different systems but tries to be as consistent as possible. Path resolution is the following:

### Windows

- Look for `data/faceengine.conf` in current working directory

### Linux

- Look for `/etc/visionlabs/faceengine.conf`
- Look for `data/faceengine.conf` in current working directory if previous options fail

The config file format is optimized for deserialization of several FSDK types:

- Int1 - scalar 32 bit integral numeric type
- Int2 - 2-d 32 bit integral numeric type (aka Vector2i, Size)
- Int3 - 3-d 32 bit integral numeric type
- Int4 - 4-d 32 bit integral numeric type (aka Rect)
- Float1 - scalar 32 bit floating point numeric type
- Float2 - 2-d 32 bit floating point numeric type (aka Vector2f)
- Float3 - 3-d 32 bit floating point numeric type
- Float4 - 4-d 32 bit floating point numeric type
- String - short null-terminated string (max. 16 characters including the null-terminator)

## Configuration file location

The location where the config file is found varies across different systems but tries to be as consistent as possible. Path resolution is the following:

### Windows:

- Look for `data/faceengine.conf` in current working directory

## Linux

- Look for `/etc/visionlabs/faceengine.conf`
- Look for `data/faceengine.conf` in current working directory if previous options fail

## Mobile platforms

- Look for “`data/faceengine.conf`” in current working directory.

## Settings

### System settings

Parameter	Description	Type	Default value
verboseLogging	Level of log verbosity. 1 - Errors, 2 - Warnings, 3 - Info, 4 - Debug.	"Value::Int1"	2
betaMode	Enable experimental features (0 - Off, 1 - On).	"Value::Int1"	0
defaultDetectorType	Detector type: FaceDetV2, FaceDetV3.	"Value::String"	FaceDetV3

Verbosity level sets the upper limit of what type of messages may be printed out by the Luna SDK. For example, if user set verboseLogging to 3, it means that Errors, Warnings and Info messages will be printed out to the console. Verbose level of 0 indicates that there are no logging messages printed out at all.

#### Example:

```
<section name="system">
  <param name="verboseLogging" type="Value::Int1" x="2" />
  <param name="betaMode" type="Value::Int1" x="0" />
  <param name="detectorType" type="Value::String" text="FaceDetV3" />
</section>
```

## Descriptor factory settings

Descriptor factory is a facility that creates descriptor extractors and matchers. Both of them utilize algorithms that require a number of coefficients (“weights”) to operate properly.

Parameter	Description	Type	Default value
model	CNN face descriptor version. Possible values: 58, 59, 60, 62.	Value::Int1	62
useMobileNet	MobileNet is faster but less accurate. Possible values: 0 - don't use mobile net version, 1 - use mobile net version.	Value::Int1	0
distance	Distance between descriptors on matching. L1 faster, L2 make better precision. Possible values: L1, L2.	Value::String	L2
descriptorCountWarningLevel	Threshold, that limits the ratio of created descriptors to the amount, defined by your license. When the threshold is exceeded, FSDK prints the warning.	Value::Float1	0.9
calcSimilarity	Enables similarity calculation during matching process. Possible values: 1 - enable, 0 - disable.	Value::Int1	1
calcDistanceSqrt	Enables calculation of the square root of distance. Possible values: 1 - enable, 0 - disable	Value::Int1	1

Parameter	Description	Type	Default value
batchCapacity	Non-public parameter. Do not change.	Value::Int1	16

Models with versions 58, 59, 60, 62 support only L2 distance.

#### Example:

```
<section name="DescriptorFactory::Settings">
  <param name="model" type="Value::Int1" x="62" />
  <param name="useMobileNet" type="Value::Int1" x="0" />
  <param name="distance" type="Value::String" text="L2" />
  <param name="descriptorCountWarningLevel" type="Value::Float1" x="
    0.9" />
  <!--Calculate similarity by default to preserve backward compatible
    -->
  <param name="calcSimilarity" type="Value::Int1" x="1" />
  <!--Calculate sqrt from distance by default to preserve backward
    compatible-->
  <param name="calcDistanceSqrt" type="Value::Int1" x="1" />
  <!--Batch capacity used by IDescriptorExtractor implementations-->
  <param name="batchCapacity" type="Value::Int1" x="16" />
</section>
```



## FaceDetV3 detector settings

Parameter	Description	Type	Default value
ScoreThreshold	Detection score threshold (RGB) in [0..1] range.	Value::Float1	0.5
ScoreThresholdNPU	Detection score threshold (RGB) in [0..1] range.	Value::Float1	0.89
ScoreThresholdIR	Detection score threshold (InfraRed) in [0..1] range.	Value::Float1	0.38
RedetectScoreThreshold	Redetect score threshold in [0..1] range.	Value::Float1	0.5
NMSThreshold	Overlap threshold for NMS in [0..1] range.	Value::Float1	0.35
NMSThresholdNPU	Overlap threshold for NMS in [0..1] range.	Value::Float1	0.35
intraNMSThreshold	Overlap threshold for intra NMS in [0..1] range.	Value::Float1	0.35
minFaceSize	Minimum face size in pixels.	Value::Int1	50
strictlyMinSize	Enforced minimum face size.	Value::Int1	0
nms	Type of NMS: mean or best.	Value::String	mean
RedetectTensorSize	Target face after preprocessing for redetect.	Value::Int1	80
	Non-public parameter. Do not change.		
RedetectFaceTargetSize	Target face size for redetect.	Value::Int1	64
	Non-public parameter. Do not change.		

Parameter	Description	Type	Default value
paddings	Extension of rectangle for RGB mode. Do not change.	Value:: Float4	see below
planPrefix	Plan prefix.	Value:: String	FaceDet_v3_a5
pyramidAlgorithm	Turn On/Off the pyramid based algorithm.	Value:: Int1	0
useLNet	Whether to use LNet or not.	Value:: Int1	1
cropPaddingAlignment	Non-public parameter. Do not change.	Value:: Int1	64
batchCapacity	Non-public parameter. Do not change.	Value:: Int1	16
concurrentBatchSubmission	Non-public parameter. Do not change.	Value:: Int1	1
detectMean	Non-public parameter. Do not change.	Value:: Float3	see below
detectSigma	Non-public parameter. Do not change.	Value:: Float3	see below
redetectMean	Non-public parameter. Do not change.	Value:: Float3	see below
redetectSigma	Non-public parameter. Do not change.	Value:: Float3	see below

All detections with a size (max of width/height) smaller than `strictlyMinSize` will be excluded from the final result.

```
<section name="FaceDetV3::Settings">
  <param name="ScoreThreshold" type="Value::Float1" x="0.5"/>    <!-- used
    for RGB mode -->
  <param name="ScoreThresholdNPU" type="Value::Float1" x="0.89"/>    <!--
    used for RGB mode -->
  <param name="ScoreThresholdIR" type="Value::Float1" x="0.38"/> <!-- used
    for InfraRed mode -->
  <param name="RedetectScoreThreshold" type="Value::Float1" x="0.5"/>
  <param name="NMSThreshold" type="Value::Float1" x="0.35"/>
```

```

<param name="NMSThresholdNPU" type="Value::Float1" x="0.35"/>
<param name="intraNMSThreshold" type="Value::Float1" x="0.35"/>
<param name="minFaceSize" type="Value::Int1" x="50" />
<param name="strictlyMinSize" type="Value::Int1" x="0" />
<param name="pyramidAlgorithm" type="Value::Int1" x="1" />
<param name="nms" type="Value::String" text="mean"/> <!-- best, mean -->
<param name="RedetectTensorSize" type="Value::Int1" x="80"/>
<param name="RedetectFaceTargetSize" type="Value::Int1" x="64"/>
<param name="useLNet" type="Value::Int1" x="1" />
<param name="paddings" type="Value::Float4" x="-0.17847424" y="
    0.06987534" z="0.18716485" w="0.05400637"/>
<param name="planPrefix" type="Value::String" text="FaceDet_v3_a5" />
<param name="cropPaddingAlignment" type="Value::Int1" x="64" />
<param name="batchCapacity" type="Value::Int1" x="16" />
<param name="concurrentBatchSubmission" type="Value::Int1" x="1" />
<param name="detectMean" type="Value::Float3" x="0.0" y="0.0" z="0.0" />
<param name="detectSigma" type="Value::Float3" x="0.0" y="0.0" z="0.0"
/>
<param name="redetectMean" type="Value::Float3" x="0.0" y="0.0" z="0.0"
/>
<param name="redetectSigma" type="Value::Float3" x="0.0" y="0.0" z="0.0"
/>
</section>

```

## FaceDetV2 detector settings

Parameter	Description	Type	Default value
FirstThreshold	1-st threshold in [0..1] range.	"Value::Float1 "	0.51385
SecondThreshold	2-nd threshold in [0..1] range.	"Value::Float1 "	0.248
ThirdThreshold	3-d threshold in [0..1] range.	"Value::Float1 "	0.76
minFaceSize	Minimum face size in pixels.	"Value::Int1"	50
strictlyMinSize	Enforced minimum face size.	"Value::Int1"	0
scaleFactor	Image scale factor.	"Value::Float1 "	0.7
padding	Extension of rectangle. Do not change.	"Value::Float4 "	see below
redetectTolerance	Redetection threshold	"Value::Int1"	0
useLNet	Whether to use LNet or not.	"Value::Int"	1

minFaceSize and scaleFactor accelerate face detection at the cost of lower recall for smaller faces.

All detections with a size (max of width/height) smaller than strictlyMinSize will be excluded from the final result.

### Example:

```
<section name="FaceDetV2::Settings">
  <param name="FirstThreshold" type="Value::Float1" x="0.51385"/>
  <param name="SecondThreshold" type="Value::Float1" x="0.248"/>
  <param name="ThirdThreshold" type="Value::Float1" x="0.76"/>
  <param name="minFaceSize" type="Value::Int1" x="50" />
  <param name="strictlyMinSize" type="Value::Int1" x="0" />
  <param name="scaleFactor" type="Value::Float1" x="0.7" />
  <param name="padding" type="Value::Float4" x="-0.20099958" y="
    0.10210337" z="0.20363552" w="0.08490226" />
  <param name="redetectTolerance" type="Value::Int1" x="0" />
  <param name="useLNet" type="Value::Int1" x="0" />
</section>
```

### **LNet**

This group of parameters is non-public. Do not change any of the parameters.

### **LNetIR**

This group of parameters is non-public. Do not change any of the parameters.

### **SLNet**

This group of parameters is non-public. Do not change any of the parameters.

## IndexBuilder settings

You can build the HNSW (Hierarchical Navigable Small Worlds) index with descriptor batches and use it for a quick search of the nearest descriptor neighbors.

Parameter	Type	Default value
numThreads	"Value::Int1"	0
construction	"Value::Int1"	2000
search	"Value::Int1"	6000
searchForEvaluation	"Value::Int1"	20

Parameters description:

*numThreads* - The number of threads to be used for building or searching. If 0 or less, use the `std::hardware_concurrency`

*construction* - Internal construction value. The greater it is, the better the graph is, but the slower the construction is. **Important:** We do not recommend that you change the value, unless you know what you are doing.

*search* - Internal search value. A greater value means a slower but more complete search. **Important:** We do not recommend that you change the value, unless you know what you are doing.

*searchForEvaluation* - Evaluation index length, a greater value means a slower but more precise evaluation.

**Example:**

```
<section name="IndexBuilder::Settings">
  <param name="numThreads" type="Value::Int1" x="0" />
  <param name="construction" type="Value::Int1" x="2000" />
  <param name="search" type="Value::Int1" x="6000" />
  <param name="searchForEvaluation" type="Value::Int1" x="20" />
</section>
```

## HumanDetector settings

This section describes human body detector parameters.

Parameter	Description	Type	Default value
ScoreThreshold	Detection score threshold (RGB) in [0..1] range.	Value :: Float1	0.5
RedetectScoreThreshold	Redetect score threshold in [0..1] range.	Value :: Float1	0.12
NMSThreshold	Overlap threshold for NMS in [0..1] range.	Value :: Float1	0.4
RedetectNMSThreshold	Non-public parameter. Do not change.	Value :: parameter.Float1	0.4
imageSize	Input image size in pixels. See below for details.	Value :: Int1	640
nms	Type of NMS: mean or best.	Value :: String	best

Parameter	Description	Type	Default value
RedetectNMS	Redetect type of NMS: mean or best.	Value :: String	mean
RedetectTensorSize	Target face after preprocessing for redetect.Non-public parameter. Do not change.	"Value :: Int1"	110
RedetectHumanTargetSize	Non-public parameter. Do not change.	Value :: Int1	85
cropPaddingAlignment	Non-public parameter. Do not change.	Value :: Int1	64
batchCapacity	Non-public parameter. Do not change.	Value :: Int1	16

The `imageSize` parameter specifies the maximum size of the image along the longest side in pixels. If the longest side of the image exceeds the specified value, it will be scaled down to that value. If both sides of the image are smaller than the specified value, no scaling will be applied.

**Example:**



```
<section name="HumanDetector::Settings">
  <param name="ScoreThreshold" type="Value::Float1" x="0.5"/>
  <param name="RedetectScoreThreshold" type="Value::Float1" x="0.12"/>
  <param name="NMSThreshold" type="Value::Float1" x="0.4"/>
  <param name="RedetectNMSThreshold" type="Value::Float1" x="0.4"/>
  <param name="imageSize" type="Value::Int1" x="640"/>
  <param name="nms" type="Value::String" text="best"/> <!-- best, mean -->
  <param name="RedetectNMS" type="Value::String" text="mean"/> <!-- best,
    mean -->
  <param name="RedetectTensorSize" type="Value::Int1" x="110"/>
  <param name="RedetectHumanTargetSize" type="Value::Int1" x="85"/>
  <param name="cropPaddingAlignment" type="Value::Int1" x="64" />
  <param name="batchCapacity" type="Value::Int1" x="16" />
</section>
```

## Head detector settings

Parameter	Description	Type	Default value
ScoreThreshold	Detection score threshold (RGB) in the [0..1] range.	"Value::Float1"	0.5
NMSThreshold	Overlap threshold for NMS in the [0..1] range.	"Value::Float1"	0.35
minHeadSize	Minimum face size in pixels.	"Value::Int1"	60
strictlyMinSize	Affects minHeadSize, see below.	"Value::Int1"	0
nms	Type of NMS: mean or best.	"Value::String"	mean
cropPaddingAlignment	Non-public parameter. Do not change.	"Value::Int1"	64
batchCapacity	Non-public parameter. Do not change.	"Value::Int1"	16
concurrentBatchSubmission	Non-public parameter. Do not change.	"Value::Int1"	1

If strictlyMinSize=1, any detections with a size less than minHeadSize are excluded from the detection results. The size of the detection is determined by the maximum value between detection width and detection height.

```
<section name="HeadDetector::Settings">
  <param name="ScoreThreshold" type="Value::Float1" x="0.5"/>
  <param name="NMSThreshold" type="Value::Float1" x="0.35"/>
  <param name="minHeadSize" type="Value::Int1" x="60" />
  <param name="strictlyMinSize" type="Value::Int1" x="0" />
  <param name="nms" type="Value::String" text="mean"/> <!-- best, mean -->
  <param name="cropPaddingAlignment" type="Value::Int1" x="64" />
  <param name="batchCapacity" type="Value::Int1" x="16" />
  <param name="concurrentBatchSubmission" type="Value::Int1" x="1" />
</section>
```

## Quality estimator settings

Quality estimator looks at several image parameters, like lightness (think overexposure), darkness (think underexposure), blurriness, illumination uniformity value, specularity value. Every float value is comparing with according threshold.

Parameter	Type	Default value
blurThreshold	"Value::Float1"	x="0.58"
lightThreshold	"Value::Float1"	x="0.58"
darknessThreshold	"Value::Float1"	x="0.52"
illuminationThreshold	"Value::Float1"	x="0.3"
specularityThreshold	"Value::Float1"	x="0.3"
usePlanV1	"Value::Int1"	x="1"
usePlanV2	"Value::Int1"	x="1"

### Example:

```
<section name="QualityEstimator::Settings">
  <param name="blurThreshold" type="Value::Float1" x="0.58"/>
  <param name="lightThreshold" type="Value::Float1" x="0.58"/>
  <param name="darknessThreshold" type="Value::Float1" x="0.52"/>
  <param name="illuminationThreshold" type="Value::Float1" x="0.3"/>
  <param name="specularityThreshold" type="Value::Float1" x="0.3"/>
  <param name="usePlanV1" type="Value::Int1" x="1" />
  <param name="usePlanV2" type="Value::Int1" x="1" />
</section>
```

**Note:** usePlanV1 toggles the Quality estimation, usePlanV2 toggles the SubjectiveQuality estimation. Note that you cannot disable both the parameters at the same time. In case you do this, you will receive the `fsdk::FSDKError::InvalidConfig` error code and the following logs:

```
[27.06.2024 12:38:59] [Error] QualityEstimator::Settings Failed to create
QualityEstimator! The both parameters: "usePlanV1" and "usePlanV2" in
section "QualityEstimator::Settings" are disabled at the same time.
```

## HeadPoseEstimator settings

The HeadPose estimator is able to compute head pose angles using raw input image data.

## AttributeEstimator settings

This estimator is able to estimate many person attributes such as:

- person's age;
- gender: male, female;

Some of estimator result values depends on threshold values listed below.

Parameter	Description	Type	Default value
netType	0 - precise type, 1 - fast type	"Value::Int1"	0
genderThreshold	gender threshold in [0..1] range.	"Value::Float1" "	0.5
adultThreshold	adult threshold in [0..1] range.	"Value::Float1" "	0.2

### Example:

```
<section name="AttributeEstimator::Settings">
  <!-- 0 - precise, 1 - fast -->
  <param name="netType" type="Value::Int1" x="0" />
  <param name="genderThreshold" type="Value::Float1" x="0.5"/>
  <param name="adultThreshold" type="Value::Float1" x="0.2"/>
</section>
```

## EyeEstimator settings

This estimator aims to determine:

- Eye state: Open, Closed, Occluded;
- Precise eye iris location as an array of landmarks;
- Precise eyelid location as an array of landmarks.

To determine more exact eye state additional auxiliary model `eye_status_estimation_*.plan` is used. You can enable this auxiliary model through config (`faceengine.conf`).

Parameter	Description	Type	Default value
useStatusPlan	0 - Off, 1 - On	"Value::Int1"	1

### Example:

```
<section name="EyeEstimator::Settings">  
  <param name="useStatusPlan" type="Value::Int1" x="1"/>  
</section>
```

## GlassesEstimator settings

The glasses estimator estimates what types of glasses, if any, a person is currently wearing. Estimation quality depends on threshold values listed below. These threshold values are set to optimal by default.

Parameter	Description	Type	Default value
noGlassesThreshold	noGlasses threshold in [0..1] range.	"Value::Float1"	1
eyeGlassesThreshold	eyeGlasses threshold in [0..1] range.	"Value::Float1"	1
sunGlassesThreshold	sunGlasses threshold in [0..1] range.	"Value::Float1"	1

### Example:

```
<section name="GlassesEstimator::Settings">
  <param name="noGlassesThreshold" type="Value::Float1" x="1"/>
  <param name="eyeGlassesThreshold" type="Value::Float1" x="1"/>
  <param name="sunGlassesThreshold" type="Value::Float1" x="1"/>
</section>
```

## OverlapEstimator settings

This estimator tells whether the face is overlapped by any object.

It returns a structure with 2 fields. The first is the value of overlapping in the range from 0.0 (is not overlapped) to 1.0 (maximum, overlapped), the second is a boolean answer.

The boolean answer depends on the threshold listed below. If the value is greater than the threshold, the answer returns true, else false.

Parameter	Description	Type	Default value
overlapThreshold	overlap threshold in [0..1] range.	"Value::Float1" "	0.01

### Example:

```
<section name="OverlapEstimator::Settings">  
  <param name="overlapThreshold" type="Value::Float1" x="0.01"/>  
</section>
```



## LivenessFPREstimator settings

Thresholds are listed below.

Parameter	Description	Type	Default value
realThreshold	threshold in [0..1] range.	"Value::Float1"	0.6

```
<section name="LivenessFPREstimator::Settings">  
  <param name="realThreshold" type="Value::Float1" x="0.6"/>  
</section>
```

## LivenessIREstimator settings

This estimator determines whether the person's face is real or fake (photo, printed image).

Image must be received from infra-red camera.

The estimator returns a boolean answer (true - is real, false - is fake).

Estimator can be used in "universal" mode. The mode is chosen depending on the camera type. Thresholds are listed below.

Parameter	Description	Type	Default value
name	universal	"Value::String"	universal
irUniversalThreshold	threshold in [0..1] range.	"Value::Float1"	0.5328

```
<section name="LivenessIREstimator::Settings">
  <param name="name" type="Value::String" x="universal"/>
  <param name="irUniversalThreshold" type="Value::Float1" x="0.5328"/>
</section>
```

## NIRLivenessEstimator settings

NIRLivenessEstimator determines whether a person's face is real or fake, such as a photo or printed image. The input image must be captured by an infrared camera. The estimator returns a boolean indicating whether the face is real (`true`) or fake (`false`).

### Configuration parameters:

Parameter	Description	Type	Default value
<code>realThreshold</code>	Threshold in [0..1] range.	<code>Value::Float1</code>	<code>0.5</code>
<code>defaultEstimatorMode</code>	Configuration of plan files usage.	<code>Value::Int1</code>	<code>2</code>

```
<section name="NIRLivenessEstimator::Settings">
  <param name="realThreshold" type="Value::Float1" x="0.5"/>
  <!-- Currently, available values to select the estimation mode are:
        1 and 2. -->
  <param name="defaultEstimatorMode" type="Value::Int1" x="2"/>
</section>
```

### Estimation modes:

The estimator can operate in two modes: a fast mode and a slower but more accurate mode. By default, it is recommended to use the more accurate mode, which is designated as mode 2.

Currently, the available values for selecting the estimation mode are: `Default`, `M1`, and `M2`. The `Default` scenario means that the mode is specified in the configuration file (see `ISettingsProvider`).

### Implementation description:

*NIRLiveness Estimation Mode Enum*

```
enum class NIRLivenessMode {
    Default, // Specified in config file.
    M1,      // M1.
    M2       // M2.
};
```

## Mouth Estimator settings

Mouth estimator predicts predominant mouth state.

Estimator accuracy depends on thresholds listed below.

FPR and TPR values are specified for 0.5 threshold

**Table 16:** “Thresholds for MouthEstimation”

Parameter	Description	Type	Default value	Threshold range	TPR	FPR
occlusionThreshold	Threshold in the [0..1] range	Value::Float1	0.5	0.4 – 0.6	0.96	0.009
smileThreshold	Threshold in the [0..1] range.	Value::Float1	0.5	0.4 – 0.6	0.97	0.04
openThreshold	Threshold in the [0..1] range.	Value::Float1	0.5	0.4 – 0.6	0.986	0.01

### Example:

```
<section name="MouthEstimator::Settings">  
  <param name="occlusionThreshold" type="Value::Float1" x="0.5"/>  
  <param name="smileThreshold" type="Value::Float1" x="0.5"/>  
  <param name="openThreshold" type="Value::Float1" x="0.5"/>  
</section>
```

## Face Occlusion Estimator Settings

The Face Occlusion estimator predicts occlusion of various face areas such as the forehead, eyes, nose, mouth, and lower face.

Estimator accuracy depends on thresholds listed below.

Parameter	Description	Type	Default	Threshold range
normalHairCoeff	Threshold for hair*.	Value::Float1	0.15	0.0 - 1.0
overallOcclusionThreshold	Overall threshold.	Value::Float1	0.14	0.0 - 1.0
foreheadThreshold	Threshold for forehead.	Value::Float1	0.1	0.0 - 1.0
eyeThreshold	Threshold for eye.	Value::Float1	0.4	0.0 - 1.0
noseThreshold	Threshold for nose.	Value::Float1	0.4	0.0 - 1.0
mouthThreshold	Threshold for mouth.	Value::Float1	0.15	0.0 - 1.0
lowerFaceThreshold	Threshold for lower face.	Value::Float1	0.2	0.0 - 1.0

### Note:

The `normalHairCoeff` parameter determines whether the presence of hair is considered when analyzing face occlusion.

- If the percentage of hair occlusion is lower than `normalHairCoeff`, hair is not taken into account in the analysis.
- If the hair occlusion percentage exceeds `normalHairCoeff`, the excess hair occlusion is added to the overall face occlusion percentage.

Example calculation:

- Overall face occlusion score without hair: 0.05
- Overall face occlusion score with hair: 0.2
- `normalHairCoeff`: 0.15

In this case, the resulting overall face occlusion score would be calculated as follows:

- Resulting score =  $0.05 + (0.2 - 0.15) = 0.1$ .

**Example:**

```
<section name="FaceOcclusionEstimator::Settings">
  <param name="normalHairCoeff" type="Value::Float1" x="0.15"/>
  <param name="overallOcclusionThreshold" type="Value::Float1" x="0.07"/>
  <param name="foreheadThreshold" type="Value::Float1" x="0.1"/>
  <param name="eyeThreshold" type="Value::Float1" x="0.15"/>
  <param name="noseThreshold" type="Value::Float1" x="0.2"/>
  <param name="mouthThreshold" type="Value::Float1" x="0.15"/>
  <param name="lowerFaceThreshold" type="Value::Float1" x="0.2"/>
</section>
```

## DeepFake Estimator settings

This estimator is designed to predict is the detected face on the input image synthetic or not.

Estimator accuracy depends on `realThreshold` and `defaultEstimatorType` listed below.

**Table 18:** DeepFakeEstimator

Parameter	Description	Type	Default value
version	The version of DeepFakeEstimator, 8 or 7	Value::Int1	8
realThreshold	Threshold in [0..1] range.	Value::Float1	0.5
defaultEstimatorType	Selects estimation mode, 1 or 2	Value::Int1	2

**DeepFake version** Two latest versions of the estimator are supported, but default SDK distribution includes only plan-files for the latest version. Plan-files for the prior version are bundled in a separate archive and need to be downloaded separately.

If `realThreshold_v<version>` for the selected version is found, it is loaded. Otherwise, if `realThreshold` is found, it is loaded. Otherwise, `realThreshold`'s default value (hardcoded in SDK) is used.

**DeepFake estimation mode** Available estimation modes are M1 and M2. If `IFaceEngine::createDeepFakeEstimator(...)` is called with `mode==DeepFakeMode::Default`, `DeepFakeEstimator::Settings::defaultEstimatorType` from `IFaceEngine::getSettingsProvider()` (@see `ISettingsProvider`) determines the mode to use. Otherwise, mode passed to `IFaceEngine::createDeepFakeEstimator(...)` determines the mode to use, while `DeepFakeEstimator::Settings::defaultEstimatorType` is ignored.

### Example:

```
<section name="DeepFakeEstimator::Settings">
  <!-- plan-files for version 7 are bundled in a separate archive -->
  <param name="version" type="Value::Int1" x="8" />

  <!-- If realThreshold_v<version> for the selected version is found,
  it is loaded. Otherwise, if realThreshold is found, it is loaded.
  Otherwise, realThreshold's default value (hardcoded in SDK) is used.
  -->
  <param name="realThreshold" type="Value::Float1" x="0.5"/>
  <!-- Currently, available values for selecting estimation scenario
  are: 1 and 2. -->
```

```
<param name="defaultEstimatorType" type="Value::Int1" x="2"/>
</section>
```

### Implementation description:

*DeepFake Estimation Mode Enum*

```
enum class DeepFakeMode {
    Default, // Specified in config file.
    M1,      // M1.
    M2       // M2.
};
```



## Medical mask estimator settings

Medical mask estimator predicts predominant mask features.

Estimator accuracy depends on thresholds listed below.

If accuracy (low FPR) is more important, TPR could be sacrificed by heightening the threshold.

Corresponding FPR and TPR values are also listed below.

### Thresholds for MedicalMaskEstimation

The below parameters specify parameter thresholds in the [0..1] range and are of the "Value : Float1" type.

#### maskThreshold

Threshold range	Default threshold	FPR range	TPR range
0.65 - 0.9	0.65	0.014 - 0.01	0.976 - 0.886

#### noMaskThreshold

Threshold range	Default threshold	FPR range	TPR range
0.65 - 0.79	0.65	0.01 - 0.005	0.94 - 0.903

#### occludedFaceThreshold

Threshold range	Default threshold	FPR range	TPR range
0.5 - 0.602	0.5	0.016 - 0.01	0.924 - 0.881

### Thresholds for MedicalMaskEstimationExtended

The below parameters specify parameter thresholds in the [0..1] range and are of the "Value : Float1" type.

#### maskExtendedThreshold

Threshold range	Default threshold	FPR range	TPR range
0.65 - 0.784	0.65	0.013 - 0.01	0.923 - 0.894

### noMaskExtendedThreshold

Threshold range	Default threshold	FPR range	TPR range
0.65 - 0.79	0.65	0.01 - 0.005	0.94 - 0.903

### maskNotInPlaceExtendedThreshold

Threshold range	Default threshold	FPR range	TPR range
0.65 - 0.85	0.65	0.009 - 0.005	0.918 - 0.833

### occludedFaceExtendedThreshold

Threshold range	Default threshold	FPR range	TPR range
0.5 - 0.602	0.5	0.016 - 0.01	0.924 - 0.881

### Example:

```
<section name="MedicalMaskEstimatorV3::Settings">
  <param name="maskExtendedThreshold" type="Value::Float1" x="0.65"/>
  <param name="noMaskExtendedThreshold" type="Value::Float1" x="0.65"
    />
  <param name="maskNotInPlaceExtendedThreshold" type="Value::Float1" x=
    ="0.65"/>
  <param name="occludedFaceExtendedThreshold" type="Value::Float1" x="
    0.5"/>
  <param name="maskThreshold" type="Value::Float1" x="0.65"/>
  <param name="noMaskThreshold" type="Value::Float1" x="0.65"/>
  <param name="occludedFaceThreshold" type="Value::Float1" x="0.65"/>
</section>
```

## RedEyeEstimator settings

Red eye estimator evaluates whether person's eyes are red in a photo or not. Red eye estimation depends on threshold value listed below. These threshold value set to optimal by default.

Parameter	Description	Type	Default value
redEyeThreshold	redEyeThreshold threshold in [0..1] range.	"Value::Float1"	0.5

### Example:

```
<section name="RedEyeEstimator::Settings">  
  <param name="redEyeThreshold" type="Value::Float1" x="0.5"/>  
</section>
```

## Depth Estimator settings

Depth estimator performs liveness check via depth image. It exposes different threshold parameters where each one of them let you configure estimator for your specific use case.

Parameter	Description	Type	Default value
maxDepthThreshold	maximum depth distance threshold in mm. Should be in [0..inf] range.	"Value::Float1"	3000
minDepthThreshold	minimum depth distance threshold in mm. Should be in [0..maxDepthThreshold] range.	"Value::Float1"	100
zeroDepthThreshold	percentage of zero pixels in input image. Threshold in [0..1] range.	"Value::Float1"	0.66
confidenceThreshold	score threshold above which person is considered to be alive. Threshold in [0..1] range.	"Value::Float1"	0.89

```
<section name="DepthEstimator::Settings">
  <param name="maxDepthThreshold" type="Value::Float1" x="3000"/>
  <param name="minDepthThreshold" type="Value::Float1" x="100"/>
  <param name="zeroDepthThreshold" type="Value::Float1" x="0.66"/>
  <param name="confidenceThreshold" type="Value::Float1" x="0.89"/>
</section>
```

## LivenessFlyingFaces Estimator settings

This estimator tells whether the person's face is real or fake (photo, printed image).

It returns a structure with 2 fields.

The first one is the value in the range from 0.0 (is not real) to 1.0 (maximum, real), the second is a boolean answer.

The boolean answer depends on the "realThreshold". If the value is greater than the threshold, the answer returns true, else false.

Parameter	Description	Type	Default value
realThreshold	Threshold in the [0..1] range.	"Value::Float1"	0.5
aggregationCoeff	Coefficient in the [0..1] range.	"Value::Float1"	0.7

### Example:

```
<section name="LivenessFlyingFacesEstimator::Settings">  
  <param name="realThreshold" type="Value::Float1" x="0.5"/>  
  <param name="aggregationCoeff" type="Value::Float1" x="0.7"/>  
</section>
```

## LivenessRGBM Estimator settings

This estimator tells whether the person's face is real or fake (photo, printed image).

It returns a structure with 2 fields.

The first one is the value in the range from 0.0 (is not real) to 1.0 (maximum, real). The second is a boolean answer.

The boolean answer depends on the "threshold". If the value is greater than the threshold, the answer returns true, else false.

This estimator work is based on background accumulation. So the "backgroundCount" parameter is the amount of the frames for the background calculation.

Other parameters are implementation specific, they are not recommended to change.

Parameter	Description	Type	Default value
backgroundCount	frames count	"Value::Int1"	100
threshold	threshold	"Value::Float1"	0.8
coeff1	Non-public parameter. Do not change.	"Value::Float1"	"0.222"
coeff2	Non-public parameter. Do not change.	"Value::Float1"	"0.222"

### Example:

```
<section name="LivenessRGBMEstimator::Settings">
  <param name="backgroundCount" type="Value::Int1" x="100"/>
  <param name="threshold" type="Value::Float1" x="0.8"/>
  <param name="coeff1" type="Value::Float1" x="0.222"/>
  <param name="coeff2" type="Value::Float1" x="0.222"/>
</section>
```

## LivenessOneShotRGBEstimator settings

This estimator tells whether the person's face is real or fake (photo, printed image). Thresholds are listed below.

Liveness protects from presentation attacks - when user tries to cheat biometric system by demonstrating fake face to the face capturing camera, but not from image substitution attacks - when fake image is sent directly to the system, bypassing the camera.

LivenessOneShotRGBEstimator supports images, which are captured on Mobile devices or Webcam (PC or laptop). Correct working of the estimator with other source images is not guaranteed.

Supported shooting mode: cooperative, which means that user must interact with the camera and look at it.

User scenarios examples: authentication in mobile application, confirmation of transactions with biometric facial verification.

Image resolution minimum requirements:

- Mobile devices - 720 × 960 px
- Webcam (PC or laptop) - 1280 x 720 px

Parameter	Description	Type	Default value
version	Estimator version, 11 or 10	"Value::Int1"	11
netType	NET version.	"Value::Int1"	0
realThreshold	Threshold in [0..1] range.	"Value::Float1"	0.5
qualityThreshold	Threshold in [0..1] range.	"Value::Float1"	0.5
calibrationCoeff_v10	Coefficient in [0..1] range.	"Value::Float1"	0.921
calibrationCoeff_v11	Coefficient in [0..1] range.	"Value::Float1"	0.8868
mobileRealThreshold	Threshold in [0..1] range.	"Value::Float1"	0.5
mobileQualityThreshold	Threshold in [0..1] range.	"Value::Float1"	0.5
mobileCalibrationCoeff_v10	Coefficient in [0..1] range.	"Value::Float1"	0.9984
mobileCalibrationCoeff_v11	Coefficient in [0..1] range.	"Value::Float1"	0.9967

Parameter	Description	Type	Default value
liteRealThreshold	Threshold in [0..1] range.	"Value::Float1 "	0.5
liteQualityThreshold	Threshold in [0..1] range.	"Value::Float1 "	0.5
liteCalibrationCoeff_v10	Coefficient in [0..1] range.	"Value::Float1 "	0.9856
liteCalibrationCoeff_v11	Coefficient in [0..1] range.	"Value::Float1 "	0.9923

Parameters ending with \_v10 apply only to version 10 of the estimator (version = 10). Parameters ending with \_v11 apply only to version 11 of the estimator (version = 11). Other parameters apply to both versions.

For example, if calibrationCoeff\_v<version> for the selected version is found, it is loaded. Otherwise, if calibrationCoeff is found, it is loaded. Otherwise, calibrationCoeff's default value (hardcoded in SDK) is used.

```
<section name="LivenessOneShotRGBEstimator::Settings">
  <param name="version" type="Value::Int1" x="11" />
  <param name="netType" type="Value::Int1" x="0" />
  <!--Parameters for backend version (netType == 0) -->
  <param name="realThreshold" type="Value::Float1" x="0.5"/>
  <param name="qualityThreshold" type="Value::Float1" x="0.5" />

  <!-- If calibrationCoeff_v<version> for the selected version is found,
  it is loaded. Otherwise, if calibrationCoeff is found, it is loaded.
  Otherwise, calibrationCoeff's default value (hardcoded in SDK) is used.
  -->
  <param name="calibrationCoeff_v10" type="Value::Float1" x="0.921"/>
  <param name="calibrationCoeff_v11" type="Value::Float1" x="0.8868"/>
  <param name="calibrationCoeff_v10_2xl" type="Value::Float1" x="0.7979"/>
  <param name="calibrationCoeff_2xl_v11" type="Value::Float1" x="0.8705"/>
  <!--Parameters for mobile version (netType == 1) -->
  <param name="mobileRealThreshold" type="Value::Float1" x="0.5"/>
  <param name="mobileQualityThreshold" type="Value::Float1" x="0.5" />
  <param name="mobileCalibrationCoeff_v10" type="Value::Float1" x="0.9984"
  />
  <param name="mobileCalibrationCoeff_v11" type="Value::Float1" x="0.9967"
  />
</section>
```



```
<!--Parameters for lite version (netType == 2) -->
<param name="liteRealThreshold" type="Value::Float1" x="0.5"/>
<param name="liteQualityThreshold" type="Value::Float1" x="0.5" />
<param name="liteCalibrationCoeff_v10" type="Value::Float1" x="0.9856"/>
<param name="liteCalibrationCoeff_v11" type="Value::Float1" x="0.9923"/>
</section>
```

## Credibility Estimator settings

Credibility estimator is trained to predict reliability of a person. It does so by returning a score value between [0;1] which will be closer to 1 if a person is more likely to be reliable and closer to 0 otherwise. Along with the output score value estimator also returns an enum value, which will give a plain answer if a person is reliable or not for a user convenience. Credibility estimator sets this enum value by comparing an output score with a reliability threshold value listed in faceengine.conf file. User can modify this threshold in CredibilityEstimator::Settings section:

Parameter	Description	Type	Default value
reliableThreshold	threshold	"Value::Float1"	0.5

### Example:

```
<section name="CredibilityEstimator::Settings">  
  <param name="reliableThreshold" type="Value::Float1" x="0.5"/>  
</section>
```

## Natural Light Estimator settings

Natural Light estimator is trained to predict natural of light on the face image.

It does so by returning a score value between [0;1] which will be closer to 1 if a light is more likely to be natural and closer to 0 otherwise.

Along with the output score value estimator also returns an enum value, which will give a plain answer if a person is reliable or not for a user convenience.

NaturalLight estimator sets this enum value by comparing an output score with a reliability threshold value listed in faceengine.conf file. User can modify this threshold in NaturalLightEstimator::Settings section:

Parameter	Description	Type	Default value
naturalLightThreshold	threshold	"Value::Float1"	0.5

### Example:

```
<section name="NaturalLightEstimator::Settings">
    <param name="naturalLightThreshold" type="Value::Float1" x="0.5"/>
</section>
```

## BlackWhite Estimator settings

Estimator checks if image is color, grayscale or infrared.

Estimator accuracy depends on thresholds listed below.

Parameter	Description	Type	Default value
colorThreshold	threshold in [0..1] range	"Value::Float1"	0.5
irThreshold	threshold in [0..1] range.	"Value::Float1"	0.5

Estimator outputs ImageColorEstimation which consists of 2 scores and color image type as enum with possible values: Color, Grayscale, Infrared.

- For color image score colorScore will be close to 1.0 and the second one infraredScore - to 0.0;
- for infrared image score colorScore will be close to 0.0 and the second one infraredScore - to 1.0;
- for grayscale images both of scores will be near 0.0.

So colorThreshold is responsible for separating Color and Grayscale images; irThreshold is responsible for separating Infrared and Grayscale images.

```
<section name="BlackWhiteEstimator::Settings">
  <param name="colorThreshold" type="Value::Float1" x="0.5"/>
  <param name="irThreshold" type="Value::Float1" x="0.5"/>
</section>
```

## Fish Eye Estimator settings

Fish Eye estimator is trained to predict fish eye effect on the face image.

It does so by returning a score value between [0;1] which will be closer to 1 if a fisheye effect is more likely to be applied to the image and closer to 0 otherwise.

Along with the output score value estimator also returns an enum value, which will give a plain answer if a person is reliable or not for a user convenience.

Fish Eye estimator sets this enum value by comparing an output score with a reliability threshold value listed in faceengine.conf file. User can modify this threshold in FishEyeEstimator::Settings section:

Parameter	Description	Type	Default value
fishEyeThreshold	threshold	"Value::Float1"	0.5

### Example:

```
<section name="FishEyeEstimator::Settings">
  <param name="fishEyeThreshold" type="Value::Float1" x="0.5"/>
</section>
```

## Background Estimator settings

This estimator is designed to evaluate the background in the original image.

Estimator accuracy depends on the thresholds listed below. The scores are defined in [0,1] range. If two scores are above the threshold, then the background is solid, otherwise the background is not solid.

Parameter	Description	Type	Default value
backgroundThreshold	threshold in [0..1] range	"Value::Float1 "	0.5
backgroundColorThreshold	threshold in [0..1] range	"Value::Float1 "	0.3

```
<section name="BackgroundEstimator::Settings">
  <param name="backgroundThreshold" type="Value::Float1" x="0.5"/>
  <param name="backgroundColorThreshold" type="Value::Float1" x="0.3"
/>
</section>
```

## Human Attribute Estimator settings

Human Attribute estimator is trained to predict a bunch of human attributes on the human image.

Human Attribute estimator sets outwear color bool values and age by comparing an output score with a corresponding threshold value listed in faceengine.conf file. User can modify this threshold in HumanAttributeEstimator::Settings section:

Parameter	Description	Type	Default value
blackUpperThreshold	threshold	"Value::Float1"	0.740
blueUpperThreshold	threshold	"Value::Float1"	0.655
brownUpperThreshold	threshold	"Value::Float1"	0.985
greenUpperThreshold	threshold	"Value::Float1"	0.700
greyUpperThreshold	threshold	"Value::Float1"	0.710
orangeUpperThreshold	threshold	"Value::Float1"	0.420
purpleUpperThreshold	threshold	"Value::Float1"	0.650
redUpperThreshold	threshold	"Value::Float1"	0.600
whiteUpperThreshold	threshold	"Value::Float1"	0.820
yellowUpperThreshold	threshold	"Value::Float1"	0.670
blackLowerThreshold	threshold	"Value::Float1"	0.700
blueLowerThreshold	threshold	"Value::Float1"	0.840
brownLowerThreshold	threshold	"Value::Float1"	0.850
greenLowerThreshold	threshold	"Value::Float1"	0.700
greyLowerThreshold	threshold	"Value::Float1"	0.690
orangeLowerThreshold	threshold	"Value::Float1"	0.760
purpleLowerThreshold	threshold	"Value::Float1"	0.890
redLowerThreshold	threshold	"Value::Float1"	0.600
whiteLowerThreshold	threshold	"Value::Float1"	0.540
yellowLowerThreshold	threshold	"Value::Float1"	0.930
adultThreshold	threshold	"Value::Float1"	0.940

### Example:

```

<section name="HumanAttributeEstimator::Settings">
    <param name="blackUpperThreshold" type="Value::Float1" x="
        0.740"/>
    <param name="blueUpperThreshold" type="Value::Float1" x="
        0.655"/>
    <param name="brownUpperThreshold" type="Value::Float1" x="
        0.985"/>
    <param name="greenUpperThreshold" type="Value::Float1" x="
        0.700"/>
    <param name="greyUpperThreshold" type="Value::Float1" x="
        0.710"/>
    <param name="orangeUpperThreshold" type="Value::Float1" x="
        0.420"/>
    <param name="purpleUpperThreshold" type="Value::Float1" x="
        0.650"/>
    <param name="redUpperThreshold" type="Value::Float1" x="
        0.600"/>
    <param name="whiteUpperThreshold" type="Value::Float1" x="
        0.820"/>
    <param name="yellowUpperThreshold" type="Value::Float1" x="
        0.670"/>

    <param name="blackLowerThreshold" type="Value::Float1" x="
        0.700"/>
    <param name="blueLowerThreshold" type="Value::Float1" x="
        0.840"/>
    <param name="brownLowerThreshold" type="Value::Float1" x="
        0.850"/>
    <param name="greenLowerThreshold" type="Value::Float1" x="
        0.700"/>
    <param name="greyLowerThreshold" type="Value::Float1" x="
        0.690"/>
    <param name="orangeLowerThreshold" type="Value::Float1" x="
        0.760"/>
    <param name="purpleLowerThreshold" type="Value::Float1" x="
        0.890"/>
    <param name="redLowerThreshold" type="Value::Float1" x="
        0.600"/>
    <param name="whiteLowerThreshold" type="Value::Float1" x="
        0.540"/>
    <param name="yellowLowerThreshold" type="Value::Float1" x="
        0.930"/>

    <param name="adultThreshold" type="Value::Float1" x="0.940"

```



```
</section> />
```

## Portrait Style Estimator settings

This estimator is designed to evaluate the status of a person's shoulders in the original image.

Estimator accuracy depends on the threshold listed below.

Parameter	Description	Type	Default value
notPortraitStyleThreshold	threshold in [0..1] range	"Value::Float1" "	0.2
portraitStyleThreshold	threshold in [0..1] range	"Value::Float1" "	0.35
hiddenShouldersThreshold	threshold in [0..1] range	"Value::Float1" "	0.2

```
<section name="PortraitStyleEstimator::Settings">
  <param name="notPortraitStyleThreshold" type="Value::Float1" x="0.2"
  />
  <param name="portraitStyleThreshold" type="Value::Float1" x="0.35"/>
  <param name="hiddenShouldersThreshold" type="Value::Float1" x="0.2"
  />
</section>
```

## HumanFace detector settings

Parameter	Description	Type	Default value
humanThreshold	Threshold in the [0..1] range.	"Value::Float1"	0.5
nmsHumanThreshold	Threshold in the [0..1] range.	"Value::Float1"	0.4
faceThreshold	Threshold in the [0..1] range.	"Value::Float1"	0.5
nmsFaceThreshold	Threshold in the [0..1] range.	"Value::Float1"	0.3
associationThreshold	Threshold in the [0..1] range.	"Value::Float1"	0.5
minFaceSize	Minimum face size in pixels.	"Value::Int1"	50
strictlyMinSize	Enforced minimum face size.	"Value::Int1"	0
batchCapacity	Non-public parameter. Do not change.	"Value::Int1"	8
cropPaddingAlignment	Non-public parameter. Do not change.	"Value::Int1"	64
useInt8	Whether to use quantized plans	"Value::Int1"	0

All detections with a size (max of width/height) smaller than `strictlyMinSize` will be excluded from the final result.

Currently, `useInt8` may be set to 1 only if `deviceClass=cpu` and `cpuClass=avx2` or `auto`. If `useInt8=1`, the detector uses the quantized plan `human_face_v<version>_cpu-avx2-int8.plan` instead of the non-quantized `human_face_v<version>_cpu-avx2.plan`. The quantized plan is a bit faster than the non-quantized one, see “CPU. HumanFaceDetector performance”. The downside is that the quantized plan offers lower accuracy and may require adjustments in settings. `minFaceSize` and `strictlyMinSize` are two primary parameters to adjust if some detections present under `useInt8=0` disappear after switching to `useInt8=1`. The quantized plan is also known to output detections whose coordinates may differ by up to 10-20 pixels on different machines.

```
<section name="HumanFaceDetector::Settings">
  <param name="humanThreshold" type="Value::Float1" x="0.5"/>
  <param name="nmsHumanThreshold" type="Value::Float1" x="0.4"/>
  <param name="faceThreshold" type="Value::Float1" x="0.5"/>
  <param name="nmsFaceThreshold" type="Value::Float1" x="0.3"/>
  <param name="associationThreshold" type="Value::Float1" x="0.5"/>
```

```
<param name="minFaceSize" type="Value::Int1" x="50"/>
<param name="strictlyMinSize" type="Value::Int1" x="0" />
<param name="cropPaddingAlignment" type="Value::Int1" x="64" />
<param name="batchCapacity" type="Value::Int1" x="8" />
<!-- If useInt8=1, then avx2-int8 plan is used whenever avx2 plan is
      requested -->
<param name="useInt8" type="Value::Int1" x="0" />
</section>
```

## Landmarks detector settings

Parameter	Description	Type	Default value
useLNet	To detect Landmarks68	"Value::Int1"	1
useSLNet	To detect Landmarks5	"Value::Int1"	1

```
<section name="LandmarksDetector::Settings">  
  <param name="useLNet" type="Value::Int1" x="1" />  
  <param name="useSLNet" type="Value::Int1" x="1" />  
</section>
```

**Note** Please pay attention, both parameters cannot be disabled at the same time. In this case, you will receive the error code (fsdk::FSDKError::InvalidConfig), and logs like below:

```
[30.08.2022 15:47:15] [Error] [FaceLandmarksDetector] Failed to create  
FaceLandmarksDetector! The both parameters: "useSLNet" and "useLNet" in  
section "LandmarksDetector::Settings" are disabled at the same time.
```

## Orientation Estimator settings

Orientation estimator detects an orientation of the input image.

Parameter	Description	Type	Default value
batchCapacity	Non-public parameter. Do not change.	Value::Int1	16

## ImageModification Estimator settings

ImageModification estimator estimates the likelihood that an image is unmodified.

Parameter	Description	Type	Default value
batchCapacity	Input batch is processed in subbatches of this size	Value::Int1	8
threshold	If score < threshold, estimation status is "modified"	Value::Float1	0.5

```
<section name="ImageModificationEstimator::Settings">
  <param name="batchCapacity" type="Value::Int1" x="8"/>

  <!-- if score < threshold, estimation status is "modified" -->
  <param name="threshold" type="Value::Float1" x="0.5"/>
</section>
```

## Crowd estimator settings

Parameter	Description	Type	Default value
defaultEstimatorType	Type of the estimator	"Value::String"	TwoNets
detectorType	Type of detector	"Value::String"	HumanDetector
minHeadSize	Target minHeadSize	"Value::Int1"	6
batchCapacity	Non-public parameter. Do not change.	"Value::Int1"	1
cropPaddingAlignment	Non-public parameter. Do not change.	"Value::Int1"	0
lowThreshold	Non-public parameter. Do not change.	"Value::Float1"	0.1
upThreshold	Non-public parameter. Do not change.	"Value::Float1"	100

There are next possible values for the defaultEstimatorType parameter:

- Single - working mode with one network usage
- TwoNets - working mode with two networks usage

```
<section name="CrowdEstimator::Settings">
  <!-- Available types are: Single, TwoNets -->
  <param name="defaultEstimatorType" type="Value::String" text="
    TwoNets"/>
  <!-- Available types are: HumanDetector, HeadDetector -->
  <param name="detectorType" type="Value::String" text="HumanDetector"
  />
  <param name="minHeadSize" type="Value::Int1" x="6"/>
  <param name="batchCapacity" type="Value::Int1" x="1"/>
  <param name="cropPaddingAlignment" type="Value::Int1" x="0"/>
  <param name="lowThreshold" type="Value::Float1" x="0.1"/>
  <param name="upThreshold" type="Value::Float1" x="100"/>
</section>
```



## LivenessDepthRGBEstimator settings

LivenessDepthRGBEstimator estimator performs liveness check via pair of depth image and RGB image. It exposes different threshold parameters where each one of them let you configure estimator for your specific use case.

Parameter	Description	Type	Default value
maxDepthThreshold	maximum depth distance threshold in mm. Should be in [0..inf] range.	"Value::Float1"	3000
minDepthThreshold	minimum depth distance threshold in mm. Should be in [0..maxDepthThreshold] range.	"Value::Float1"	100
zeroDepthThreshold	percentage of zero pixels in input image. Threshold in [0..1] range.	"Value::Float1"	0.66
confidenceThreshold	score threshold above which person is considered to be alive. Threshold in [0..1] range.	"Value::Float1"	0.5

```
<section name="LivenessDepthRGBEstimator::Settings">
  <param name="maxDepthThreshold" type="Value::Float1" x="3000"/>
  <param name="minDepthThreshold" type="Value::Float1" x="100"/>
  <param name="zeroDepthThreshold" type="Value::Float1" x="0.66"/>
  <param name="confidenceThreshold" type="Value::Float1" x="0.5"/>
</section>
```

## DepthLivenessEstimator settings

DepthLivenessEstimator performs liveness check for a face depth warp image. It exposes different threshold parameters that let you configure the estimator for your specific use case.

Parameter	Description	Type	Default value
maxDepthThreshold	maximum depth distance threshold in mm. Should be in [0..inf] range.	"Value::Float1"	3000
minDepthThreshold	minimum depth distance threshold in mm. Should be in [0..maxDepthThreshold] range.	"Value::Float1"	100
zeroDepthThreshold	percentage of zero pixels. Should be in [0..1] range.	"Value::Float1"	0.66
confidenceThreshold	score threshold above which person is considered to be alive. Should be in [0..1] range.	"Value::Float1"	0.5

- Pixels greater than **maxDepthThreshold** are zeroed out/excluded from estimation.
- Pixels less than **minDepthThreshold** are zeroed out/excluded from estimation.
- If the percentage of zero pixels in input image is greater than **zeroDepthThreshold**, the whole image is zeroed out, even if there are some good pixels, and the estimator returns a score close to 0.

```
<section name="DepthLivenessEstimator::Settings">
  <param name="maxDepthThreshold" type="Value::Float1" x="3000"/>
  <param name="minDepthThreshold" type="Value::Float1" x="100"/>
  <param name="zeroDepthThreshold" type="Value::Float1" x="0.66"/>
  <param name="confidenceThreshold" type="Value::Float1" x="0.5"/>
</section>
```

## FightsEstimator settings

FightsEstimator estimator performs a fight detection on the several frame sequences from the target video. It exposes different parameters where each one of them let you configure estimator for your specific use case.

Parameter	Description	Type	Default value
batchSize	count of frames in one sequence (batch)	"Value::Int1"	5
minBatchCount	minimum sequences (batches) count	"Value::Int1"	5
cropSize	internal crop size. do not change it!	"Value::Int1"	224
threshold	score threshold above which the video is considered to contain a fight. Threshold in [0..1] range.	"Value::Float1"	0.5
scoreNorm	normalization parameter. do not change it!	"Value::Float1"	1.8

```
<section name="FightsEstimator::Settings">
  <param name="batchSize" type="Value::Int1" x="5"/>
  <param name="minBatchCount" type="Value::Int1" x="5"/>
  <param name="cropSize" type="Value::Int1" x="225"/>
  <param name="threshold" type="Value::Float1" x="0.5"/>
  <param name="scoreNorm" type="Value::Float1" x="1.8"/>
</section>
```

## Runtime settings

Runtime configuration file provides parameters that user can tweak to achieve optimal performance of their app.

The name of runtime configuration file is `runtime.conf` and its placed in `data` directory. Its settings are described below:

Parameter	Type	Default value
<code>cpuClass</code>	"Value::String"	"auto"
<code>deviceClass</code>	"Value::String"	"cpu"
<code>numThreads</code>	"Value::Int1"	-1
<code>verboseLogging</code>	"Value::Int1"	0
<code>numComputeStreams</code>	"Value::Int1"	4
<code>programCacheSize</code>	"Value::Int1"	128
<code>defaultGpuDevice</code>	"Value::Int1"	0
<code>cpuHighWatermark</code>	"Value::String"	4294967296
<code>gpuHighWatermark</code>	"Value::String"	2147483648

Parameters description:

*cpuClass* - class of cpu by supported instructions - `cpu`, `sse4`, `avx`, `avx2`, `arm`, `auto`.

*deviceClass* - execution device type - `cpu`, `gpu`.

*numThreads* - number of worker threads. Default: number of CPU logical cores.

*verboseLogging* - level of log verbosity. 1 - Errors, 2 - Warnings, 3 - Info, 4 - Debug.

*numComputeStreams* - number of streams; Increases performance, but works only with new versions of NVIDIA drivers (375.82, 384.59 and more recent). Don't increase it with older version of NVIDIA driver.

*programCacheSize* - maximum number of Program objects in cache. Should be less than 10000.

*defaultGpuDevice* - default GPU device number.

*cpuHighWatermark* - CPU high-watermark options for runtime. The default value - 4294967296, means 4 Gbyte.

*gpuHighWatermark* - GPU high-watermark options for runtime. The default value - 2147483648, means 2 Gbyte.

Verbosity level sets the upper limit of what type of messages may be printed out. For example, if user set verboseLogging to 3, it means that Errors, Warnings and Info messages will be printed out to the console. Verbose level of 0 indicates that there are no logging messages printed out at all.

In case of GPU usage the numThreads value should be at least == 2 or -1. If this requirement is violated, further behavior is undefined.

Increasing the programCacheSize increases memory usage and potentially improves performance. Be careful, too large a value of this parameter can lead to a crash due to insufficient memory.

**Example:**

```
<section name="Runtime">
  <param name="cpuClass" type="Value::String" text="auto" />
  <param name="deviceClass" type="Value::String" text="cpu" />
  <param name="numThreads" type="Value::Int1" x="-1" />
  <param name="verboseLogging" type="Value::Int1" x="0" />
  <param name="numComputeStreams" type="Value::Int1" x="4" />
  <param name="programCacheSize" type="Value::Int1" x="128" />
  <param name="defaultGpuDevice" type="Value::Int1" x="0" />
  <param name="cpuHighWatermark" type="Value::String" text="4294967296" />
  <param name="gpuHighWatermark" type="Value::String" text="2147483648" />
</section>
```

*Note:* Setting <param name="numThreads" type="Value::Int1" x="-1"/> means that will be taken the maximum number of available threads. This number of threads is equal to according number of available processor cores.

*Note:* Setting <param name="defaultGpuDevice" type="Value::Int1" x="-1"/> means disable GPU runtime initialization. Set it only with deviceClass == cpu.