画像認識を用いた
自転車盗難防止アプリの作成

２０１９年度　ＯＩＳＡ技術研究会
Ｐｙｔｈｏｎ部会　Ｂチーム

２０１９年１２月１１日

【部会員】

三浦　伊織	株式会社オーイーシー
片山　達郎	株式会社オーイーシー
森本　真理子	モバイルクリエイト株式会社
大坊　亮一	株式会社オルゴ
佐藤　純平	株式会社シーエイシー
藤澤　雄太	株式会社富士通九州システムズ

アジェンダ

0．はじめに
1．今回作成したシステムの概要
2．今回使用した技術要素
3．顔認証について
4．デモンストレーション
5．学習／認証について
6．まとめ

０．はじめに

Pythonで何が出来るか？

・Webアプリ開発（Youtube ,Instagram , Dropbox）
・自動でデータ処理や分析
・自動でWebサイトのデータを収集（スクレイピング）
・機械学習、ディープラーニング
・画像認識・画像処理　　etc・・・

０．はじめに

Pythonで何をするか？

・音声認識
・普段の業務等の自動化（RPA）
・ディープラーニングによるオセロのAI
・データの可視化
・スクレイピングによる
　　プロ野球のデータ収集&分析

画像認識を用いた自転車盗難防止アプリの作成を目指す！

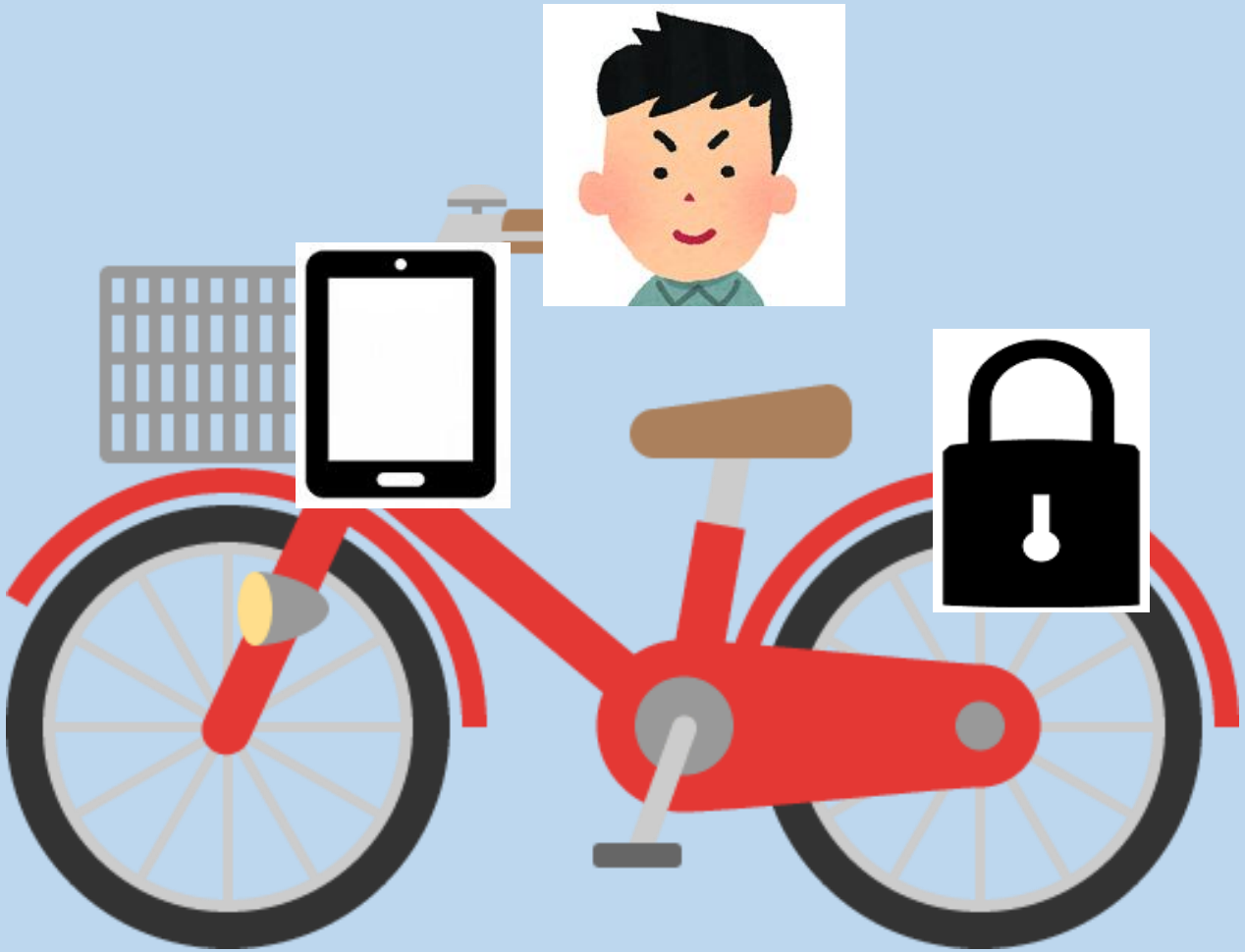
1．今回作成したシステムの概要

今回作成した部分

認証結果

顔画像

画像認証



2．今回使用した技術要素

OpenCV

画像処理・画像解析および機械学習等の機能を持つ
オープンソースのライブラリ
Python , C/C++ , Java , MATLAB用として
公開されている

Haar-like特徴

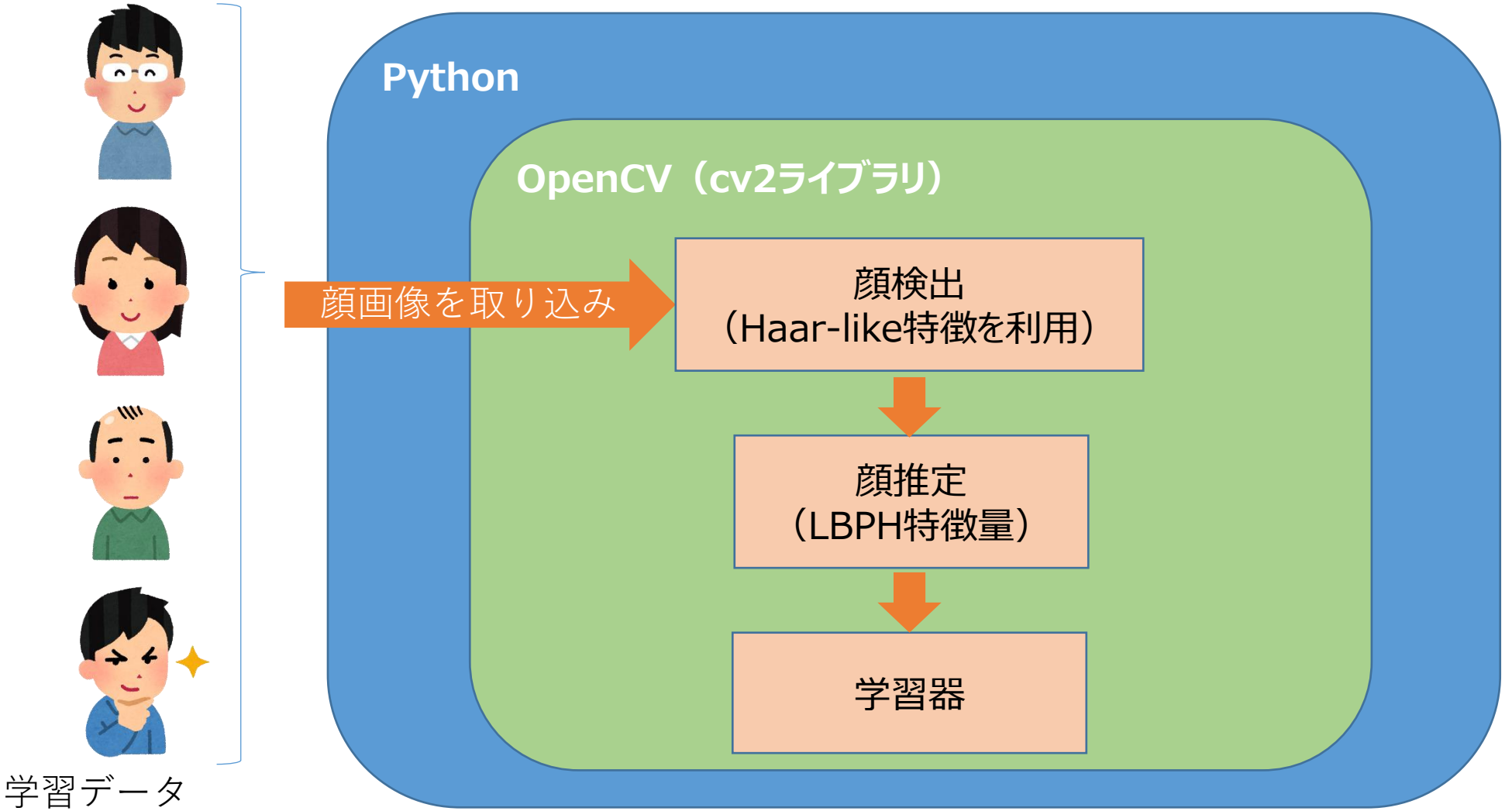
顔の部位ごとのパターンをまとめた特徴情報

LBPH特徴量

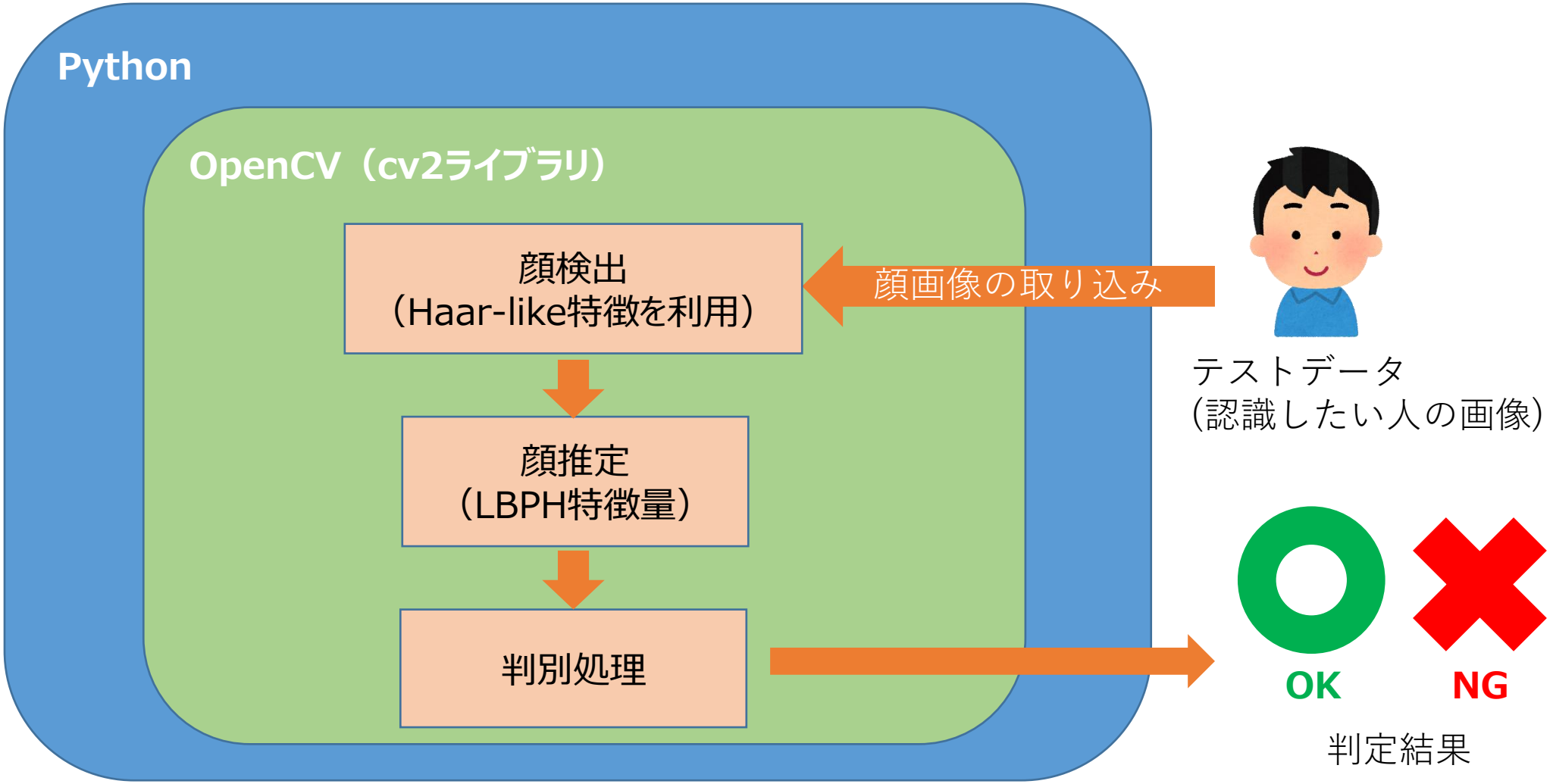
OpenCVで計算できる、輝度(明暗)の差を基準とした特徴量

※特徴量…学習データにどのような特徴があるかを数値化したもの

2．今回使用した技術要素
顔認証の仕組み（学習フェーズ）

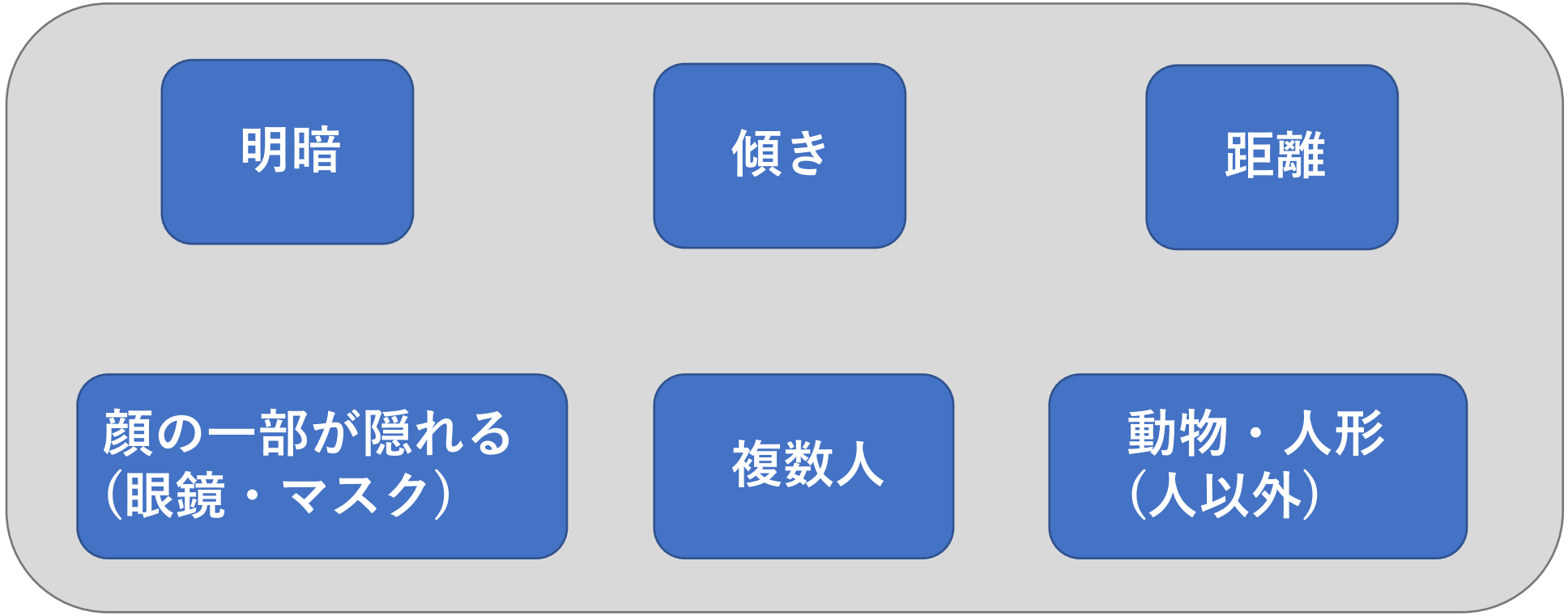


2．今回使用した技術要素
顔認証の仕組み（予測フェーズ）



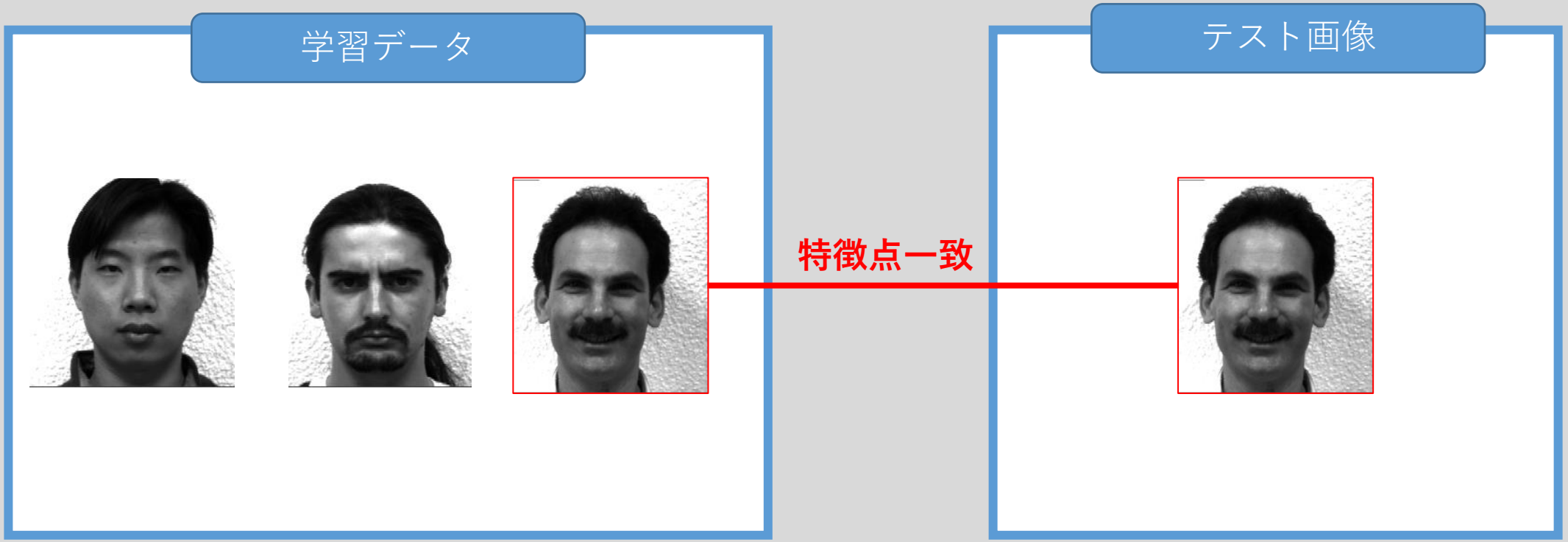
３．顔認証について

- どういったデータが認識するのか？



３．顔認証について

誤差：検証する顔がどのくらい本人に近いかを数値で表したもの
→0に近いほど、本人である確率が高い



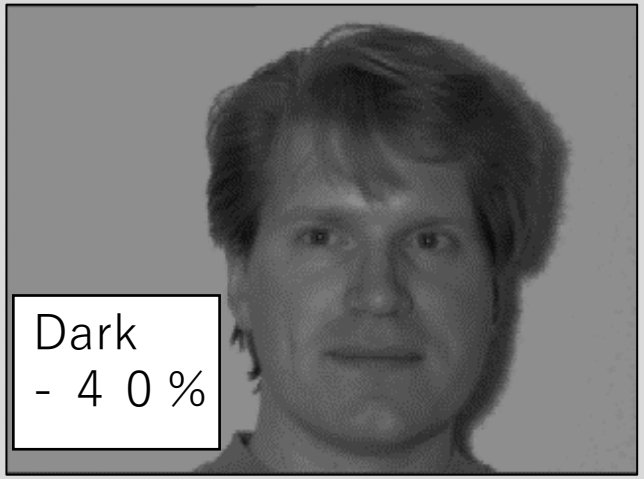
３．顔認証について

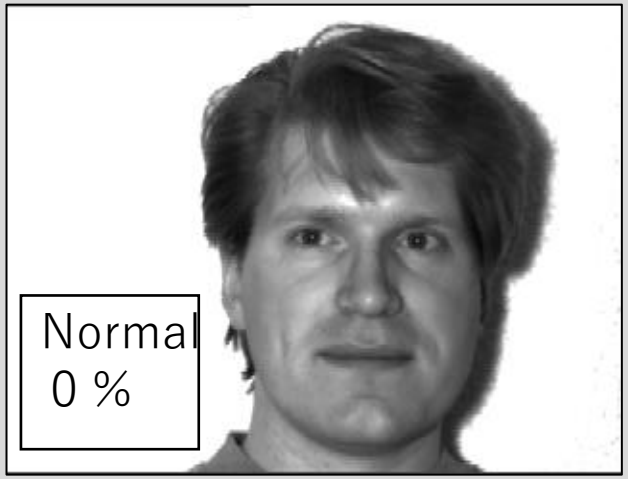
＜**明暗**＞

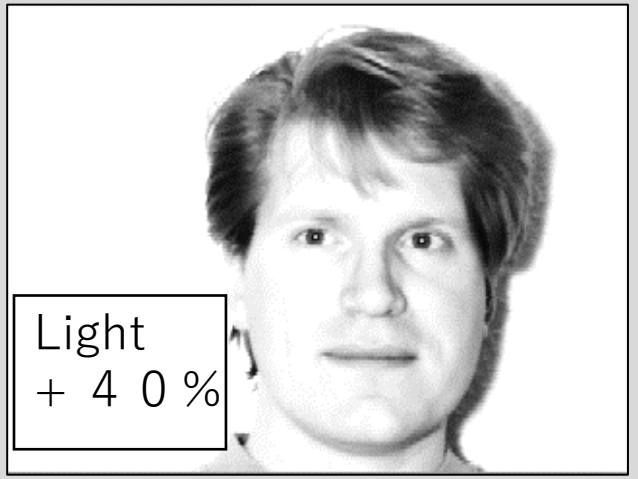
- 画像を三枚用意(Dark・Normal・Light)

  結果：学習データにすべての明暗のパターンがある時が

  一番誤差が小さくなった







３．顔認証について

<傾き>

- 一枚の画像を15度毎に回転させた場合

結果：0度～±15度までは顔を認識できているが、

それ以上の顔については認識できない

→これは、顔認識に使用しているHaar-like特徴量が

<u>正常状態の顔をパターン元</u>としているからと考えられる。

3．顔認証について

＜**距離**＞

- 様々な距離の学習データの場合

  結果：学習データと同一の距離での誤差は小さくなる

  ※学習データを増やしても誤差は変化しない

  という結果になった

例)カメラとの距離を
40cm離して撮影した場合

40cmで撮影したデータの誤差：小
※ほかの距離のものは誤差：大

３．顔認証について

＜顔の一部が隠れる場合＞

- 眼鏡・マスク・サングラスの場合

結果：マスクは認識されないが、眼鏡と
サングラス(色の薄いもの)は認識できる
※ただし、誤差は60,70前後

３．顔認証について

＜**複数人**＞

- 2~15人の画像を用いて、それぞれの顔をちゃんと認識できるか

  結果：最大15人でもすべての顔の認識ができる

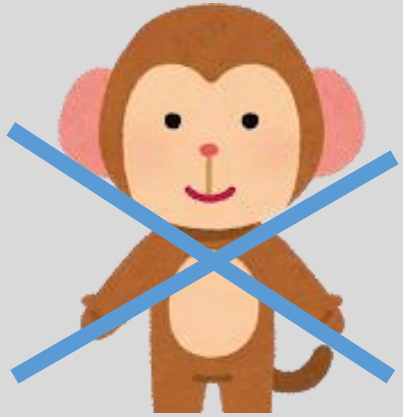
※ただし、顔の系統が
同じような人は
正しく認識できない

３．顔認証について

<人以外>

• 動物の場合

結果：顔認識自体できなかった



・人形（リカちゃん人形・こけしなど）の場合

結果：顔認識するが、誤差はかなり大きい

→誤差：80、90など

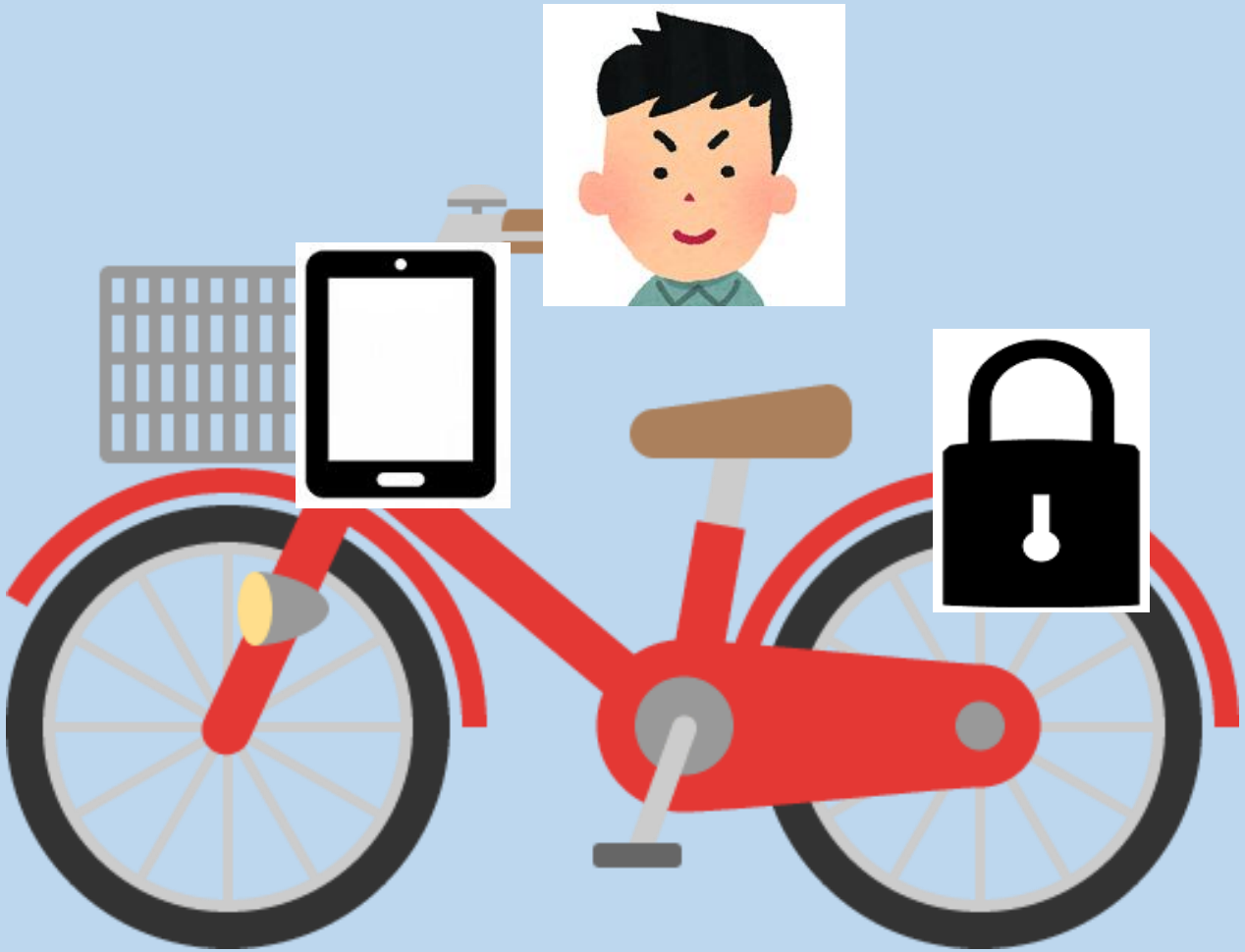
4．デモンストレーション

今回作成した部分

認証結果

顔画像

画像認証



５．検証－学習／認証について

- 学習の検証
  → 学習させる顔画像による誤差の変化を検証
  → 今回の自転車盗難防止アプリでは

- 認証の検証
  →実際にいろんな場所で認証を行い誤差を検証

5．検証－学習について

学習させる顔画像による誤差の変化を検証

・誤差が小さくなった検証ケース
  ・いろんな明度の顔画像を学習させた場合
  ・いろんな距離の顔画像を学習させた場合
  ・眼鏡や帽子を着用した顔写真を混ぜて学習させた場合

・誤差に変化が無かったケース
  ・認証者を複数学習させた場合
    → 他認証者と誤認するリスクあり
  ・距離が離れている顔データ(画像が小さいもの)
  ・画質が悪いもの
  ・顔が正面を向いていない(横を向いているなど)
    → 認識しづらい

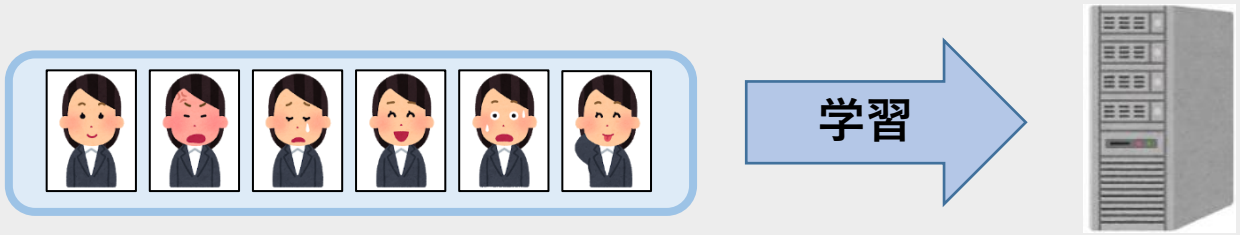
⇒ 幅広いパターンの画像を学習させることで誤差が小さくなる

５．検証－学習について

今回の自転車盗難防止アプリでは

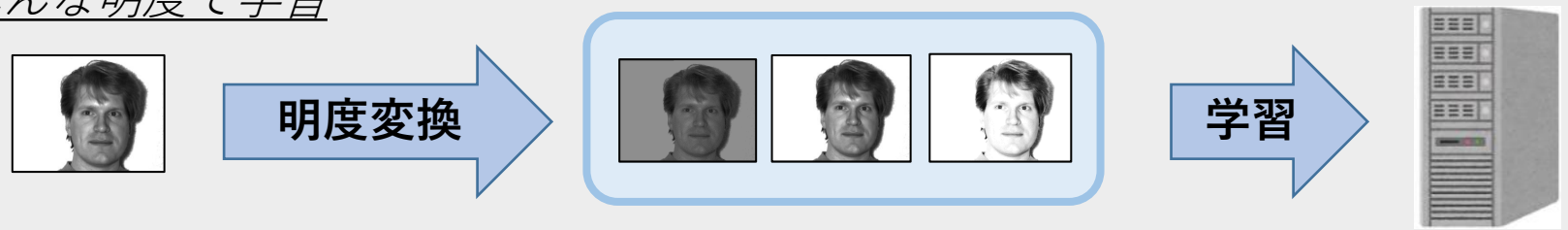
- 同じ表情をしていない事が想定される

色んな表情で学習



- 場所・時間の違いにより明暗の違いが想定される

色んな明度で学習



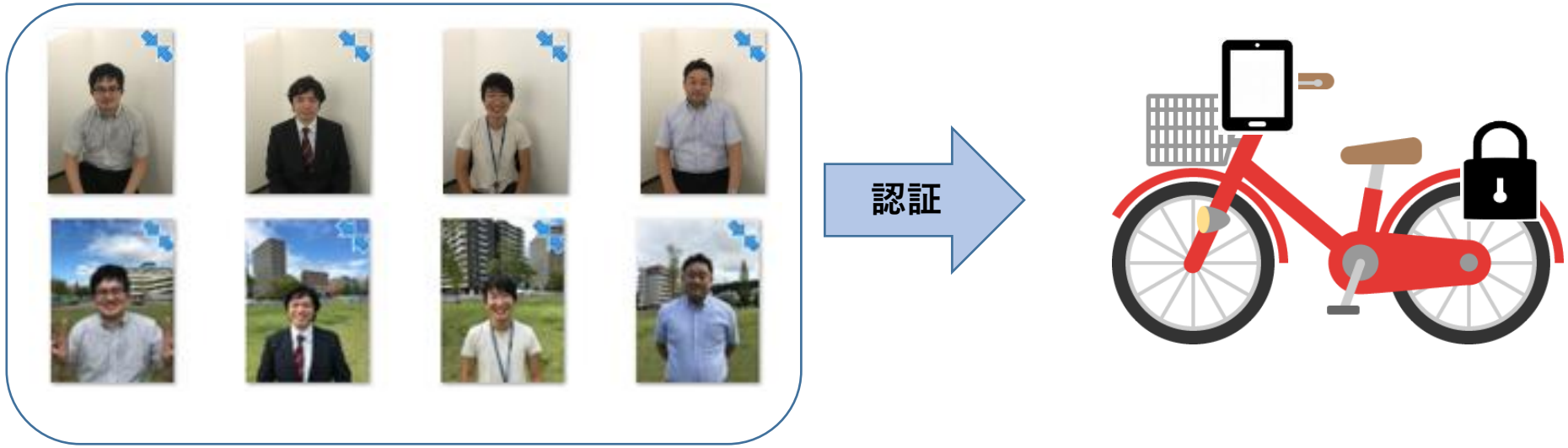
１枚の画像から複数の学習データを作成

各メンバー１０枚ずつの写真を使用し、学習を実施した

5．検証－認証について

実際にいろんな場所で認証を行い誤差を検証

- 室内外の視点で認証実施



⇒ 室内外問わず認証できたが、他人を認証してしまうケースもあった（→閾値を調整）

6．まとめ

誤認識を防ぐには？

⇒閾値を厳しくする

問題　・閾値を厳しくすると本人でも認識されない

・数百枚の画像を集めるのは大変

問題解決する為に

①プログラムの精度を上げる

・取込データの明暗調整

・複数の認証方法を組み合わせて判断

②鍵が開かない時の代替方法を準備する

・物理的な鍵やパスワード