So you want a pet robot: A camera-based leader-follower system for small mobile robots

Camille Monnier*, Stan German, Andrey Ostapchenko

Boston Image Processing and Computer Vision Group (BIPCVG)

March 14, 2013
Motivation

- Most fielded robots demand a human operator’s full attention
  - Even for mundane tasks such as travel
- Robots/UGVs are typically controlled via bulky control units
  - Soldiers are already burdened with equipment
Goals

- Can we automate simple tasks?
  - Let the operator continue to do what he/she does best
- Can we design more intuitive/lighter-weight controls?
A Solution: Robotic Mule

- Develop a “mule-like” capability for semi-autonomous control
  - Follows a designated leader
  - Responds to natural commands (e.g., voice, gestures)
  - Platform agnostic
Prior/Ongoing Work

- Multiple approaches are being developed
- Typically platform-centric, tightly integrated
- Specialized sensors and/or user-worn equipment is common
Challenges

- Reduce reliance on specialized sensors, platform-specific hardware, maps, GPS
- Track leader through long occlusions, dynamic scenes
- Provide intuitive, hands-free controls
Our Approach

- Monocular Unmanned Leader-Follower (MULE-F)
  - Software
  - Single camera
  - Small computer
Development Platform

- Dragon Runner SUGV (QinetiQ North America)
- Manta-125C Gig-E 1280x960 Camera (Allied Vision Tech)
- Small PC (Intel Core i3 2.5GHz CPU)
Software Components

- Pedestrian Detection
  - Where are the people?

- Leader Tracking
  - Which one is the leader?

- Gesture Recognition
  - What does the leader want me to do?
Pedestrian Detection: Sliding Window Detection

- A (usually fixed-size) window is passed over the image
Sliding Window Detection

- A (usually fixed-size) window is passed over the image
Sliding Window Detection

- A (usually fixed-size) window is passed over the image
- Features are extracted from each window and analyzed
Sliding Window Detection

- A (usually fixed-size) window is passed over the image
- Features are extracted from each window and analyzed
- This is repeated at multiple scales

Scale can vary wildly!
Pedestrian Detection

- **Sliding-window detection framework**

**OFFLINE**
- Collect & Annotate Data
- Extract Features
- Train Classifier
  \[ H(x) = \sum \alpha_i h_i(x) \]

**ONLINE**
- Capture Image Stream
- Extract Features
- Evaluate Classifier
Data Collection

- Goal is to build and validate a data-driven detector

- We need examples of the object (foreground) and everything else (background)

- Foreground:

- Background:

- Sources:
Feature Extraction for Pedestrian Detection

- Haar-like Wavelets
- Edge Density/Edge Orientation Histograms
- Color Distribution
- Local Binary Patterns, Color Self-Similarity ...
Classification

- We need to learn a function that maps our *representation* into a *class label*.

- We use a classifier to do this
  - **Adaboost**, SVM, Logistic Regression, Random Forests, Naïve Bayes, Nearest Neighbor...

- Adaboost is a popular choice for detection
  - Simple to implement
  - Robust
  - Fast
  - Implicitly selects discriminative features
  - Can be *partially* evaluated on “easy” examples (a “cascade”)
Pedestrian Detection
Leader Tracking

- The system must reliably track and recognize the leader:
  - Through occlusions by confusers (e.g., passers-by)
  - Around corners and obstacles

- Two complementary types of information are available
  - Kinematic (velocity, acceleration constraints)
  - Appearance (clothing color, texture; equipment; shape, gait)
Leader Tracking: Matching Detections to Tracks

- Tracks are modeled using a Particle Filter framework
  - Each track is modeled using many samples of possible states ("particles")
  - Particles are weighted according to likelihood (supporting evidence)
  - Robust to nonlinear changes in state
  - Supports multiple modes

- Match likelihoods (particle weights) are computed using both motion and appearance data (e.g., Breitenstein et al., 2010):

\[
    w_i \propto \alpha p_d + \beta k_t + \gamma c_t
\]

- \( w_i \) is the weight of the\( i \)-th particle
- \( \alpha, \beta, \gamma \) are weights
- \( p_d \) is detector confidence
- \( k_t \) is kinematic agreement
- \( c_t \) is appearance agreement

Leader Tracking: Online Appearance Learning

- **Goal:** learn what differentiates the leader from others
  - Clothes, equipment, hair

- **Extract shape, texture, and color features**
  - HSV histograms
  - Color covariance, moments
  - HOG, LBP
  - Features extracted over overlapping grid

- **Adaboost (one-vs-all) for learning appearances**
  - Implicitly selects features
  - Lightweight
  - Resists overfitting
Multiple Target Tracking
Leader Tracking

- 1700m loop in park
  - Multiple bystanders, repeated occlusions
  - Variable lighting, motion blur
Gesture Recognition: Overview

- Goal is to enable intuitive/lightweight control
- We started with two basic commands
  - “Follow me” / “Start”
  - “Stop”
We can use the same framework and features already applied to detect and track pedestrians

- Collect and annotate a large dataset of examples of gestures/non-gestures
- Extract features
- Train a classifier (e.g., Adaboost) to recognize gestures
- At run-time, run classifier on image of tracked leader
Gesture Recognition: Data Collection

- Record video of volunteers performing gestures
  - Vary posture, viewing angle
  - Vary lighting, clothing, background
- Use pedestrian detector to bootstrap annotation process
Gesture Recognition

- 98.7% mean recognition rate (~1% false positives)
- Classifier responses are temporally filtered
- Gesture recognition requires ~4ms/frame

Stop gesture in red
Start gesture in green
Questions?