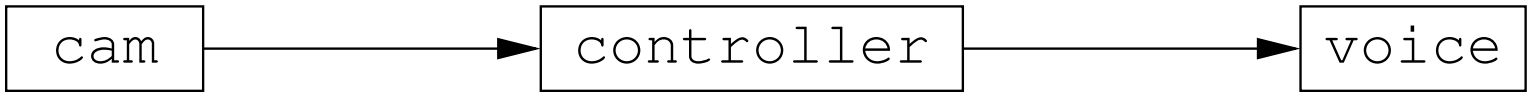
# ChessEye

Louis Mandel  Jérôme Siméon  Philippe Suter

Palaver
July 12, 2016

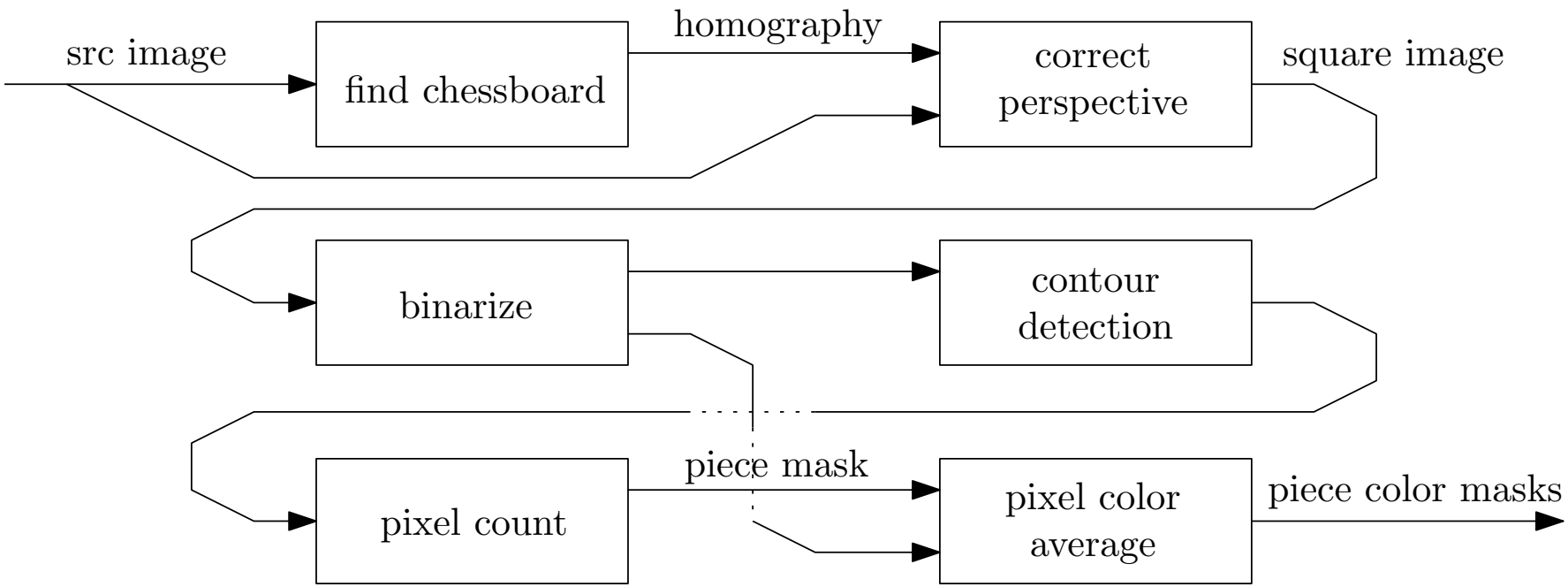
# Architecture



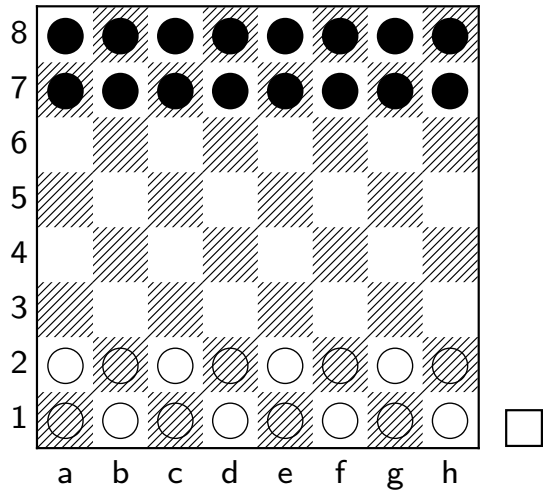

```
$ python cam/chesseye.py --src=1 \
    | controller/controller -kibbitz 1 \
    | python voice/speak.py
```

- cam: Python + OpenCV
- controller: OCaml + ReactiveML + OChess
- voice: Python + python-chess + Watson

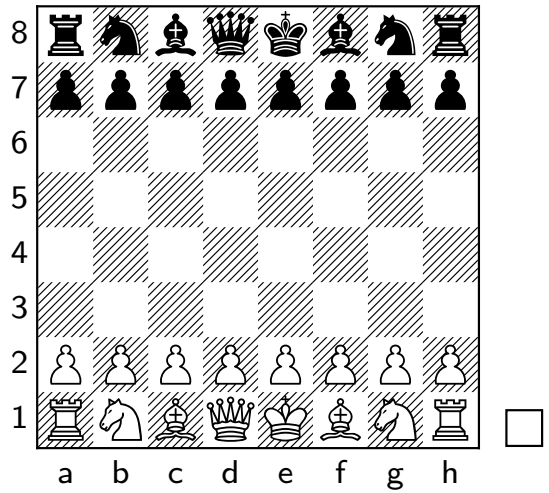
# Vision

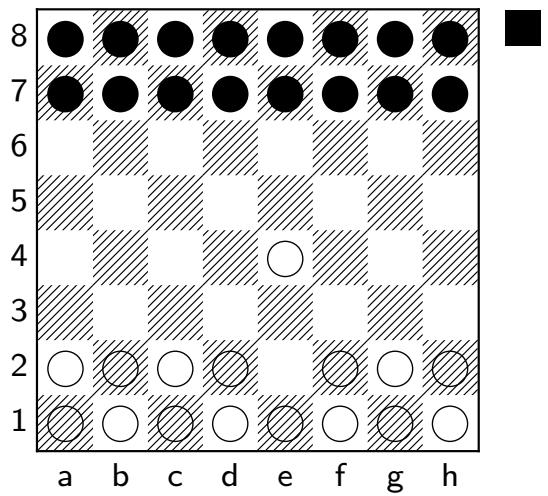

src image
find chessboard
homography
correct perspective
square image
binarize
contour detection
pixel count
piece mask
pixel color average
piece color masks

# Position of mask: Initial



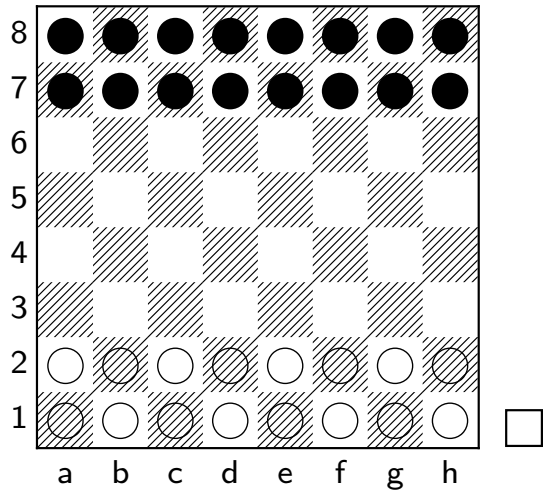

$\Longrightarrow$

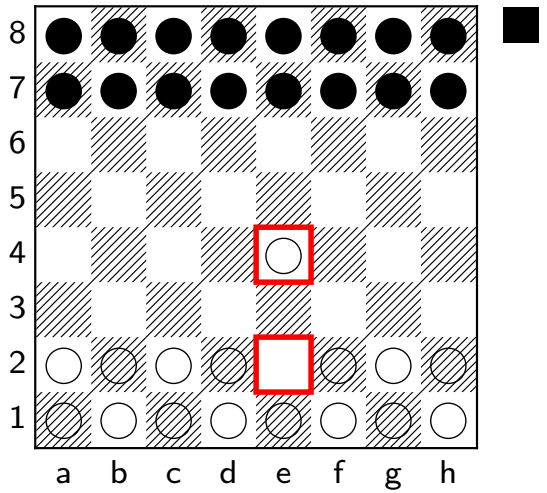


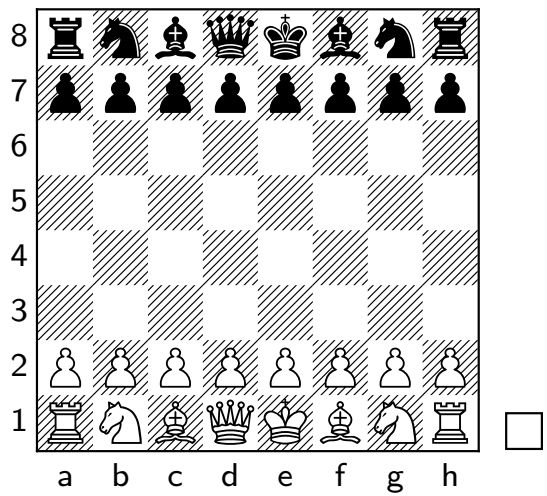



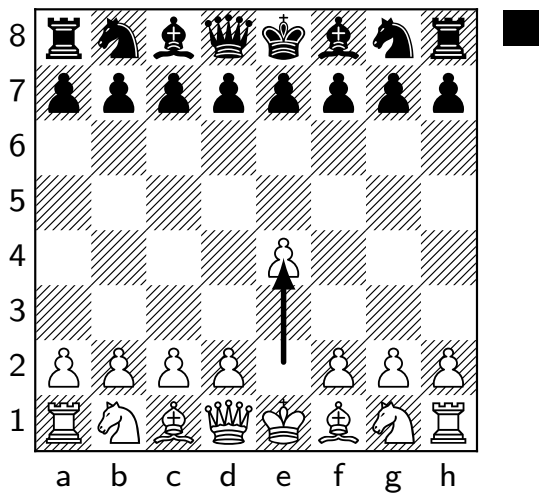

$\Longrightarrow$ ?

# Position of mask: Move



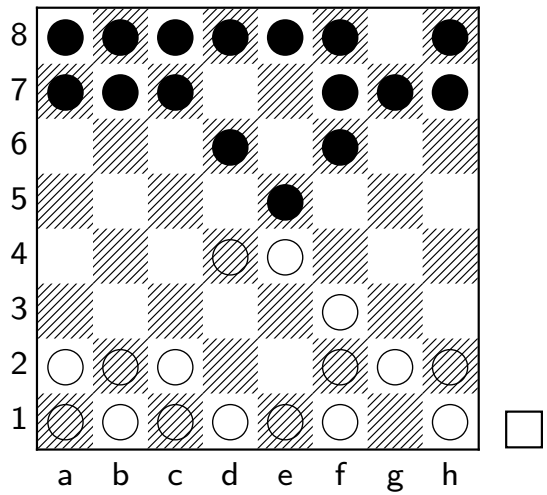



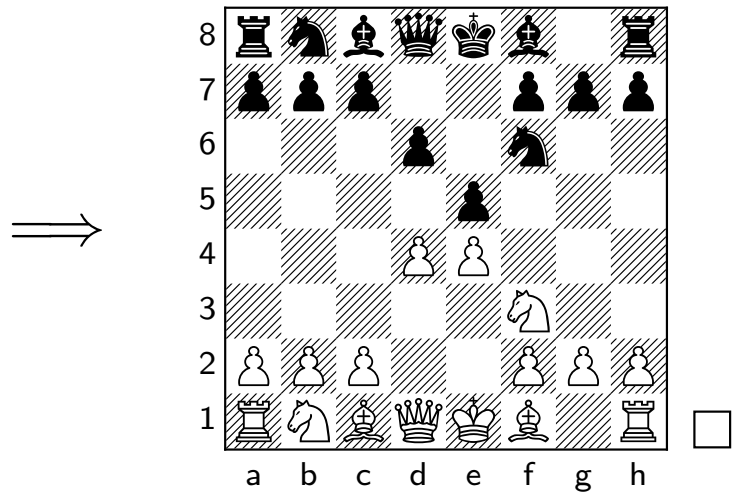



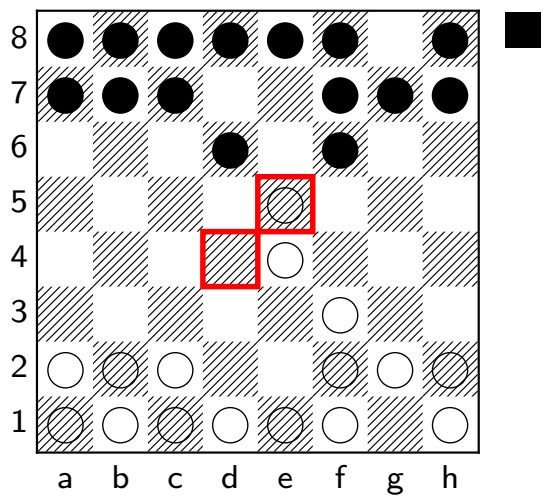



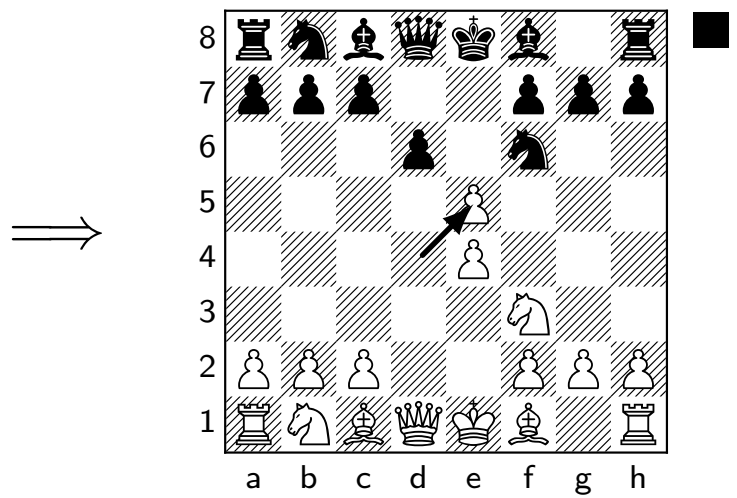
# Position of mask: Capture

**1 e2–e4 e7–e5 2 ♘g1–f3 ♘g8–f6 3 d2–d4 d7–d6**



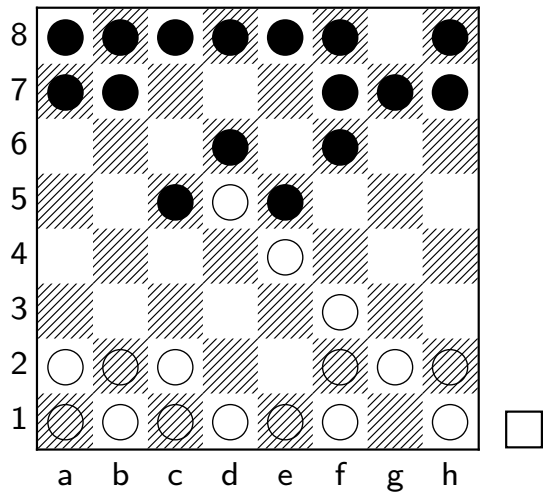



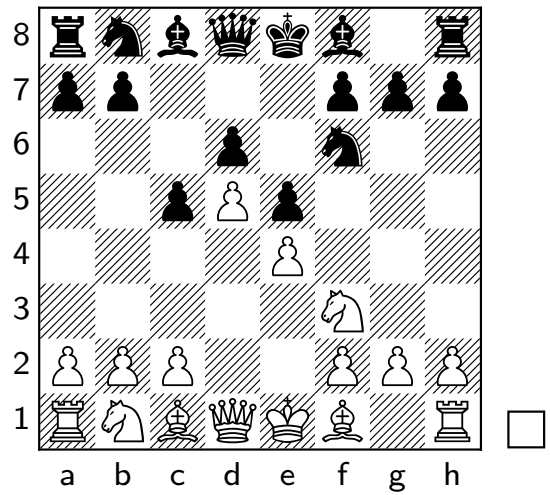



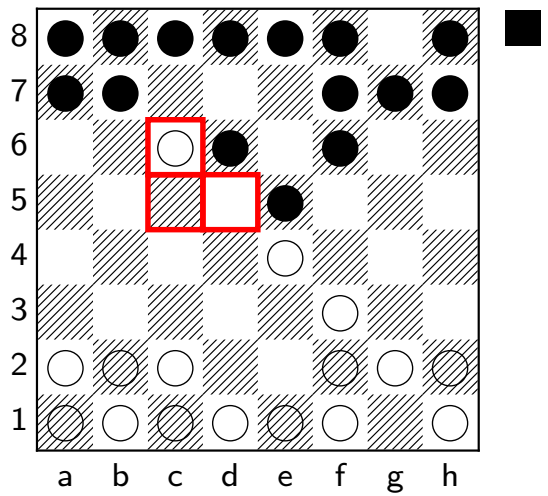



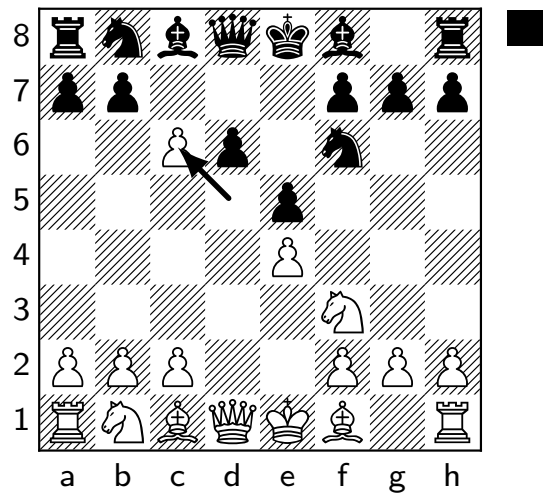
# Position of mask: Capture *en passant*

**1 e2–e4 e7–e5 2 ♘g1–f3 ♘g8–f6 3 d2–d4 d7–d6 4 d4–d5 c7–c5**



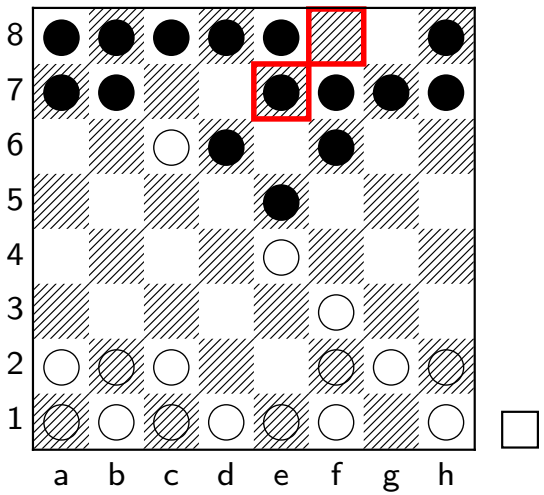



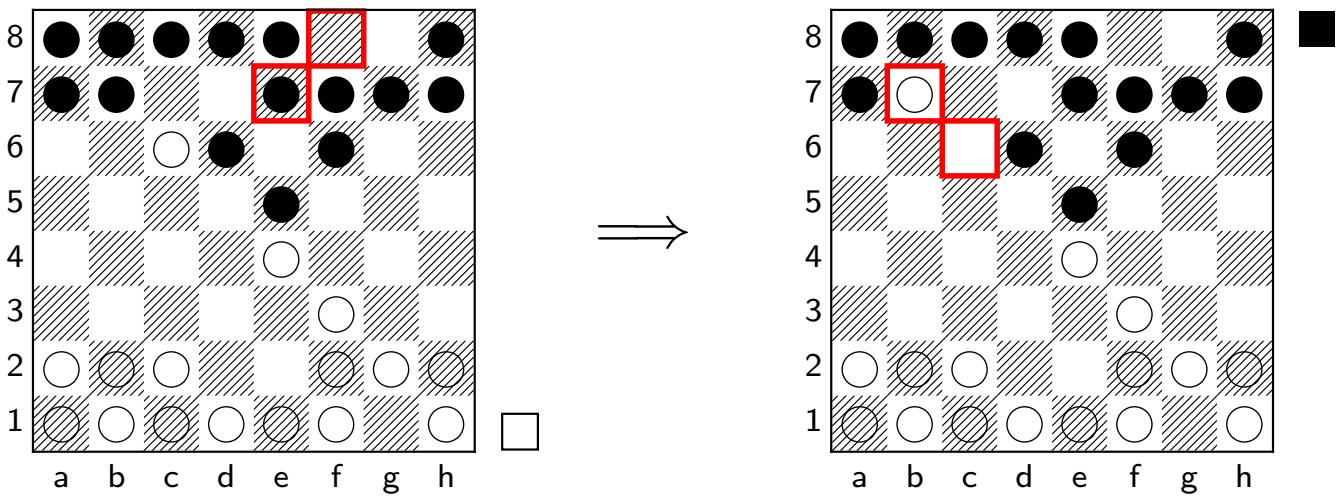
# Position of mask: Castling & Promotion

**5. . . ♗e7**



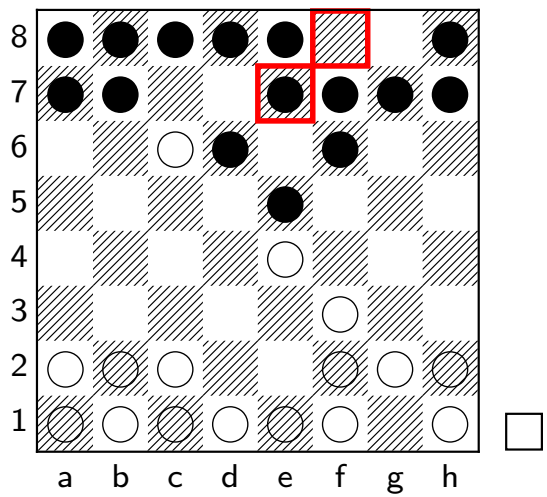
# Position of mask: Castling & Promotion

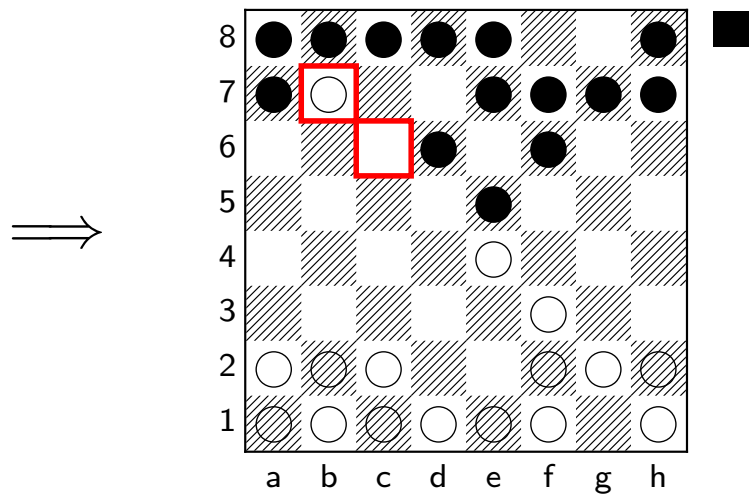
**5. . . ♗e7 6 c×b7...**

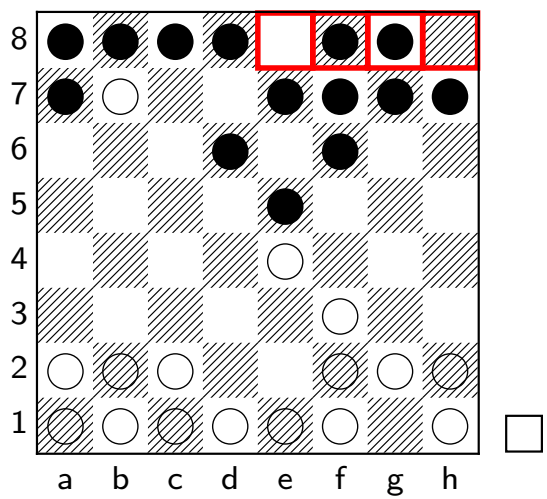


# Position of mask: Castling & Promotion

**5. . . ♗e7 6 c×b7 O-O**

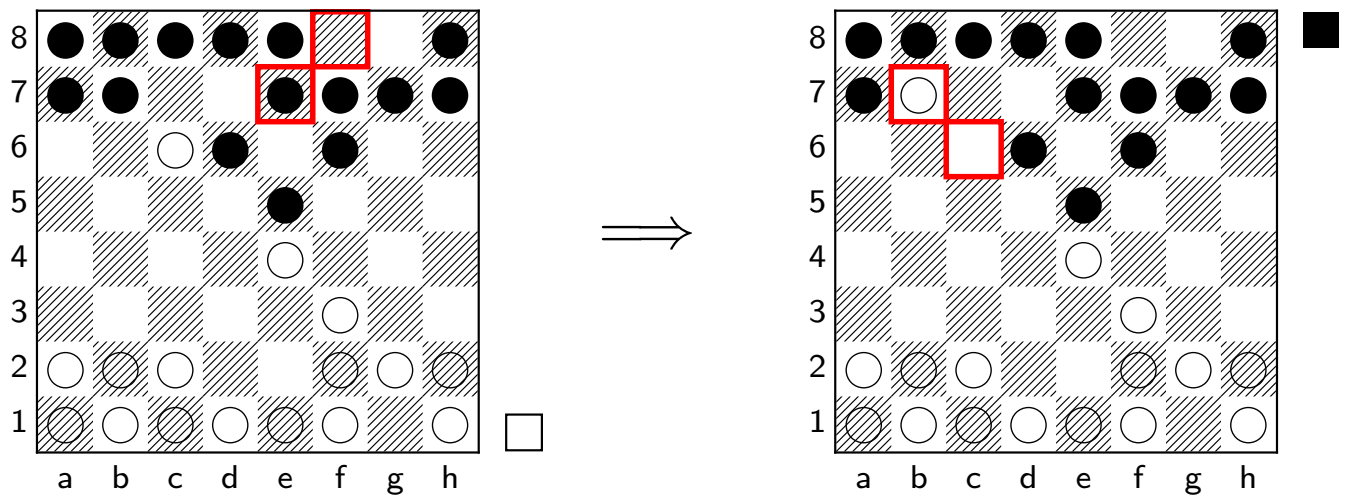


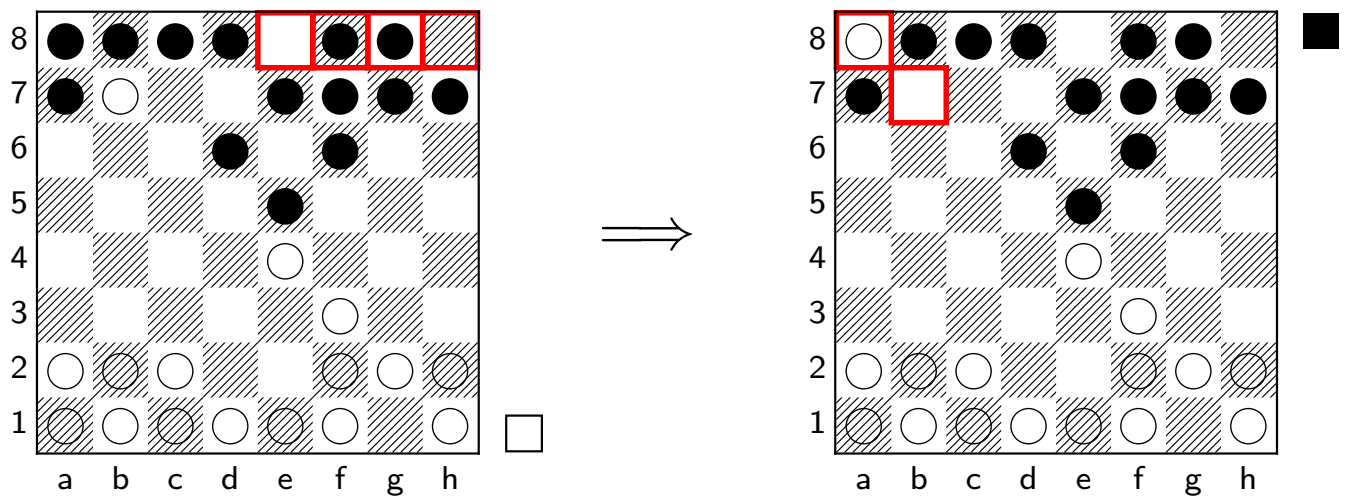



# Position of mask: Castling & Promotion

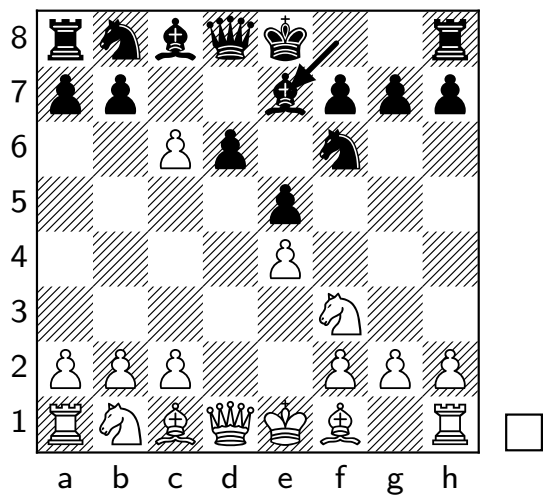
**5. . . ♗e7 6 c×b7 O-O 7 b×a8♕...**

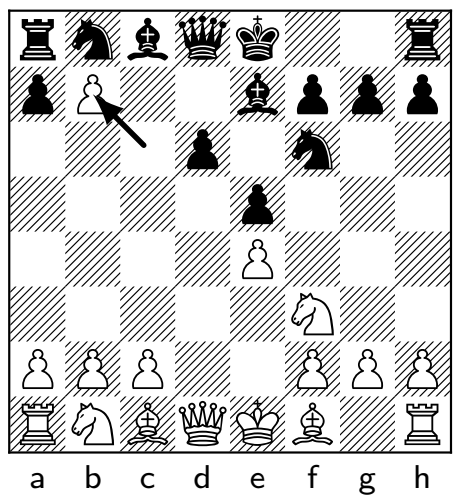


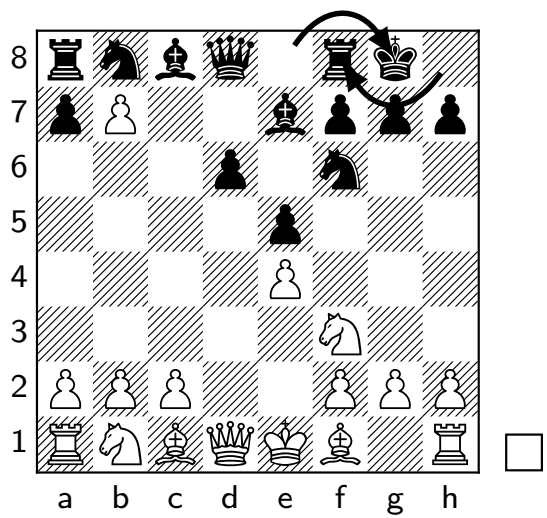



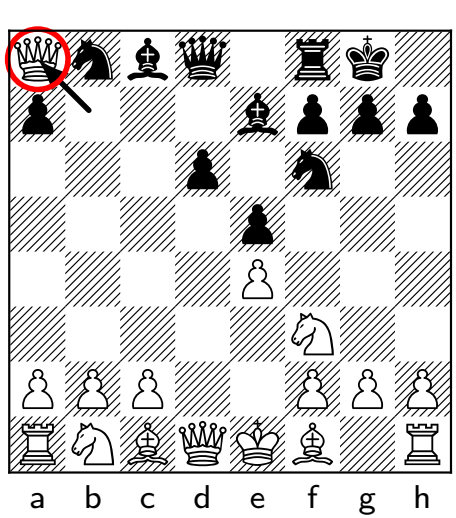
# Position of mask: Castling & Promotion

**5. . . ♗e7 6 c×b7 O-O 7 b×a8♕...**



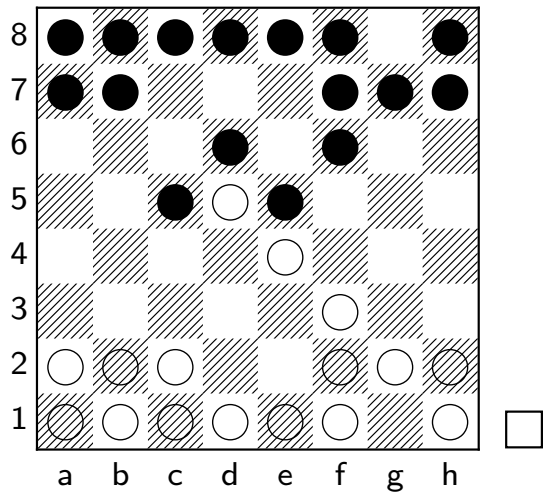



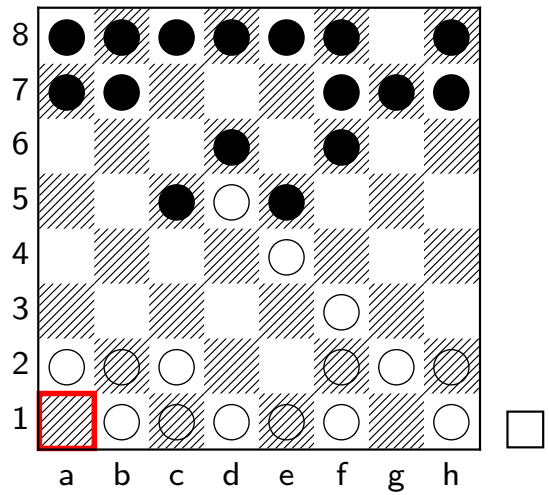



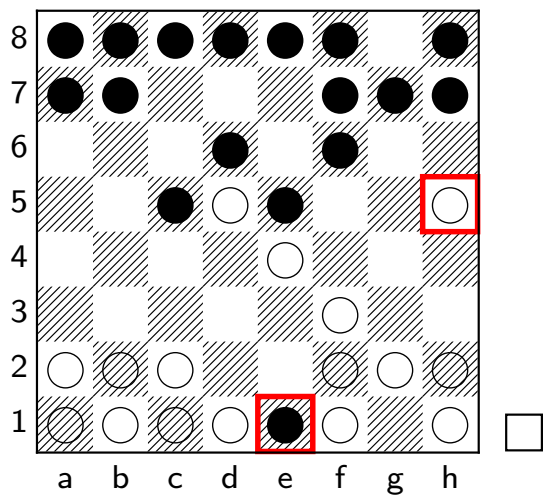



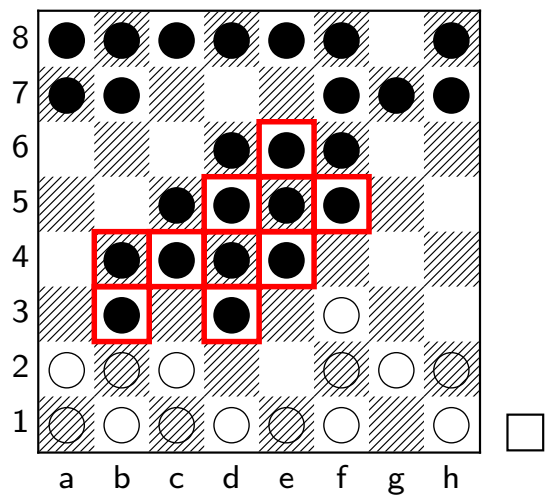
# Position of mask: Noise

**1 e2–e4 e7–e5 2 ♘g1–f3 ♘g8–f6 3 d2–d4 d7–d6 4 d4–d5 c7–c5**

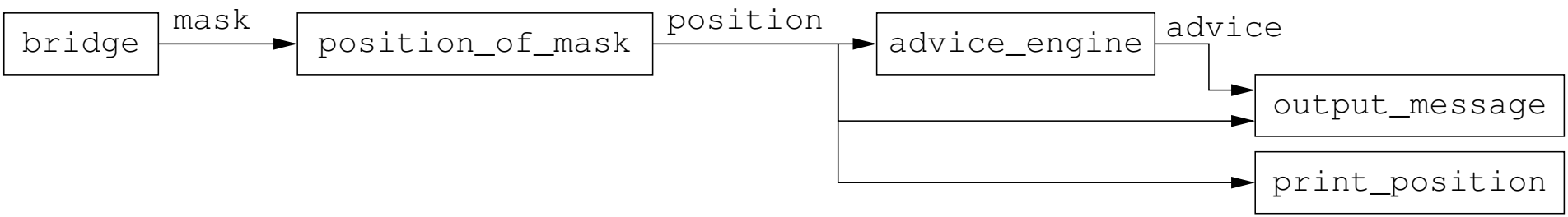


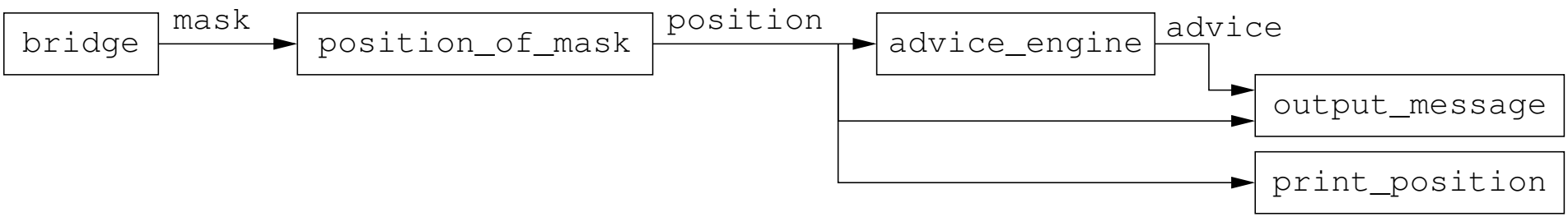



# Controller: architecture



# Controller: architecture

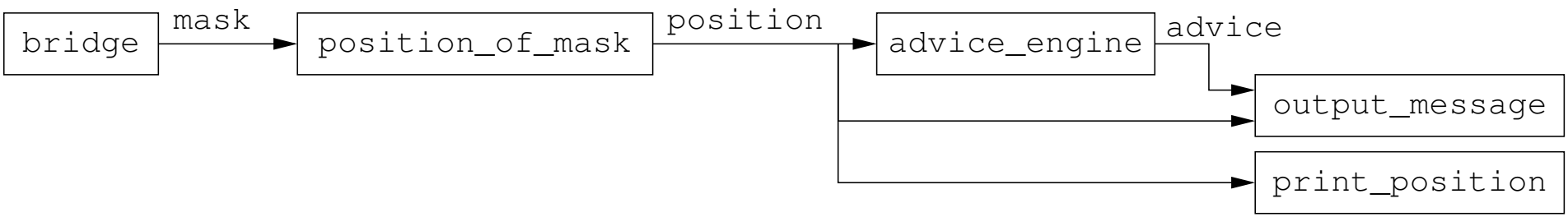



```
signal mask default None gather keep_last in
signal position default None gather keep_last in
signal advice default None gather keep_last in
begin
  run Bridge.bridge_async mask ||
  run position_of_mask mask position Ochess.init_position ||
  run advice_engine position advice ||
  run output_messages position advice ||
  run print_position position
end
```

# Controller: example of process

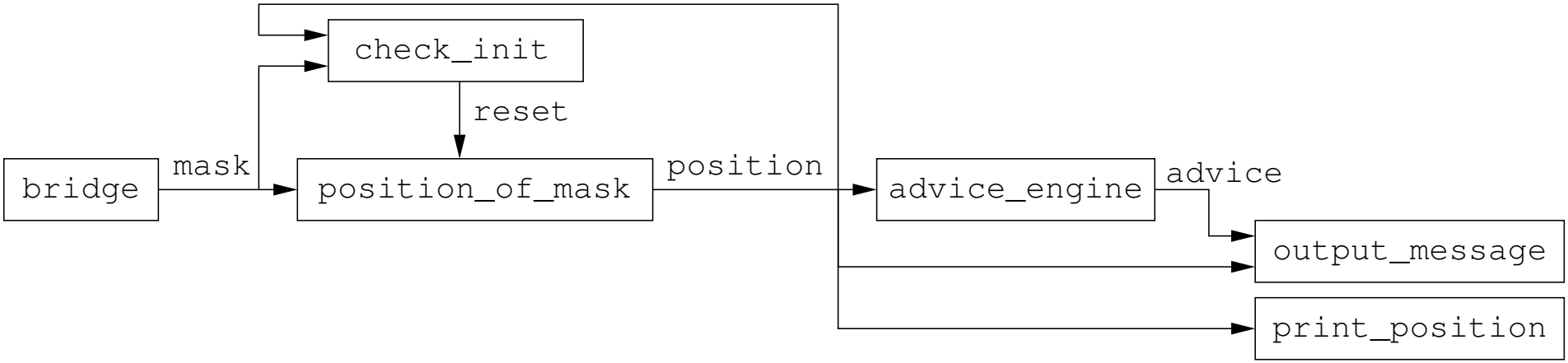
```
let process output_messages position advice =
  loop
    await position (Some (move, pos)) in
    print_endline ("MOVD "^(Ochess.long_string_of_move move pos));
    match Ochess.game_status pos with
    | Ochess.Win color -> print_endline "ENDG checkmate"
    | Ochess.Draw -> print_endline "ENDG stalemate"
    | Ochess.Play _ -> ()
  end
  ||
  loop
    await advice (Some (pos, smove)) in
    let msg = Ochess.long_string_of_smove pos smove in
    print_endline ("KIBB "^msg)
  end
```

# Why ReactiveML?




```
signal mask default None gather keep_last in
signal position default None gather keep_last in
signal advice default None gather keep_last in
begin
  run Bridge.bridge_async mask ||
  run position_of_mask mask position Ochess.init_position ||
  run advice_engine position advice ||
  run output_messages position advice ||
  run print_position position
end
```

# Controller with reset




```
signal mask default None gather keep_last in
signal position default None gather keep_last in
signal advice default None gather keep_last in
signal reset default () gather (fun () () -> ()) in
begin
  run Bridge.bridge_async mask ||
  loop
    do
      run position_of_mask mask position Ochess.init_position
    until reset done
  end ||
  run check_init position mask reset ||
  ...
end
```

# Voice

- `MOVD "rnbqkbnr/pppppppp/8/8/8/8/PPPPPPPP/RNBQKBNR w KQkq - 0 0" "e2e4"`

# Voice

- `MOVD "rnbqkbnr/pppppppp/8/8/8/8/PPPPPPPP/RNBQKBNR w KQkq - 0 0" "e2e4"`

```
def pronounce_move(fen_str, uci_str):
    # Move data re-interpreted by python-chess
    board = chess.Board(fen_str)
    move = board.parse_uci(uci_str)

    s = board.san(move) # standard algebraic notation, e.g: "Qxc7"

    s = s.replace("K", "king ")
    s = s.replace("Q", "queen ")
    s = s.replace("R", "rook ")
    s = s.replace("B", "bishop ")
    s = s.replace("N", "knight ")
    s = s.replace("x", " takes ")
    s = s.replace("+", ", check!")
    s = s.replace("#", ", checkmate!")

    say(san)
```

# Deliverable

- Source code: `https://github.com/chesseye/chesseye`
- Web site: `http://devpost.com/software/chesseye`
- Talk: `https://youtu.be/bYtGw61YLRk`
  - background video: `https://vimeo.com/165765674`

# Great business impact

**Gilad Penn** @freeslugs · May 8

holy shit - dope hack! when can i buy one :P **#chesseye** #techcrunchdisrupt #opencv