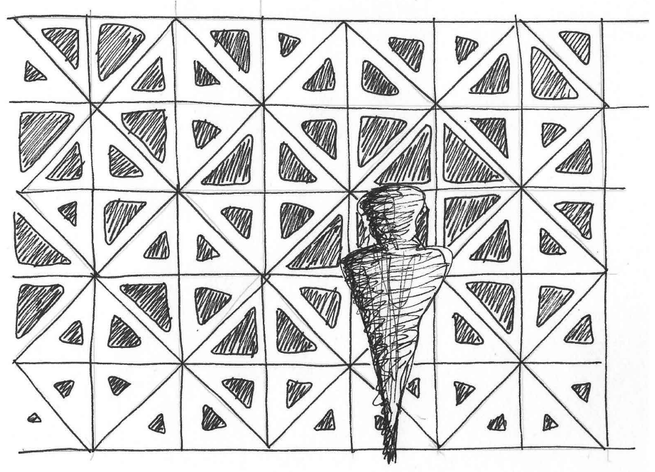
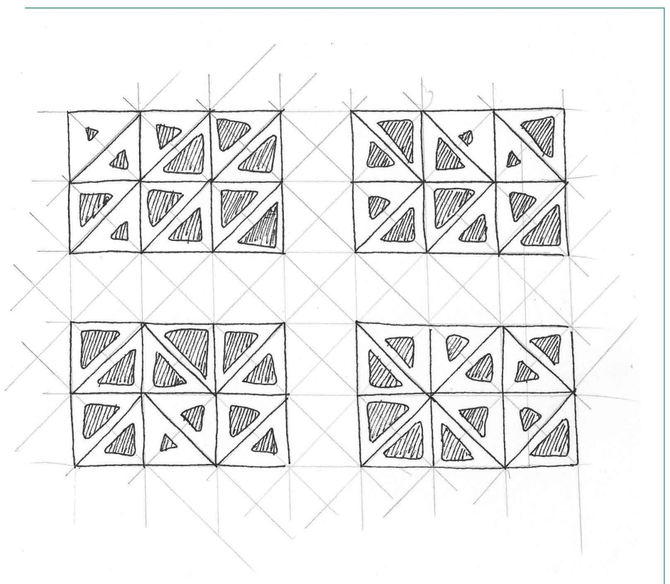
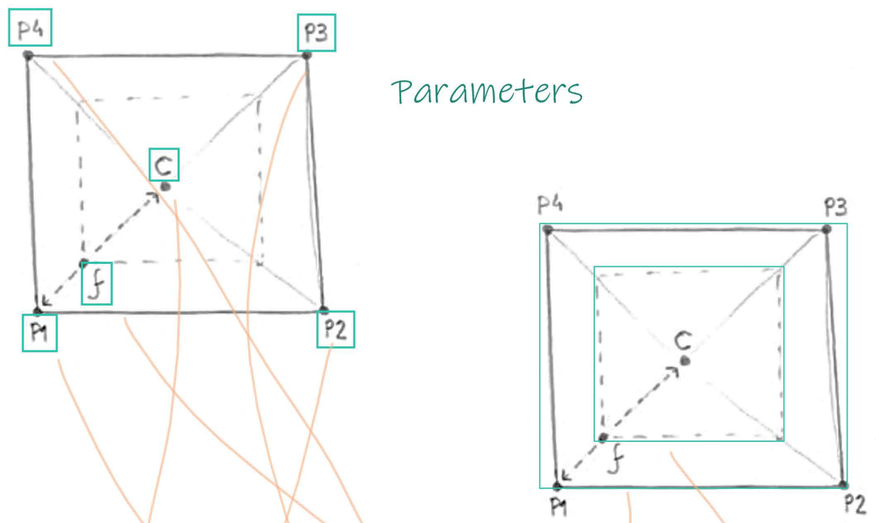
INTERACTIVE DESIGN ENVIRONMENT



DRAWING RECOGNITION



Integration of drawings in the algorithm
Detection and classification of elements in the drawings
Automatic translation to machine-written text

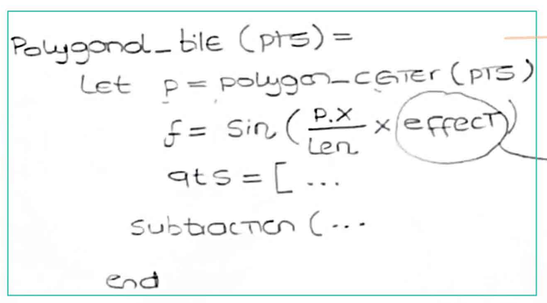


Parameters

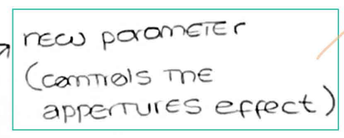
Geometry

```
squared_tile(p1, p2, p3, p4, f) =
    let pts = [p1, p2, p3, p4]
        c = polygon_center(pts)
        qts = [intermediate_loc(c, p, f) for p in pts]
      subtraction(surface_square(pts), surface_square(qts))
    end
```

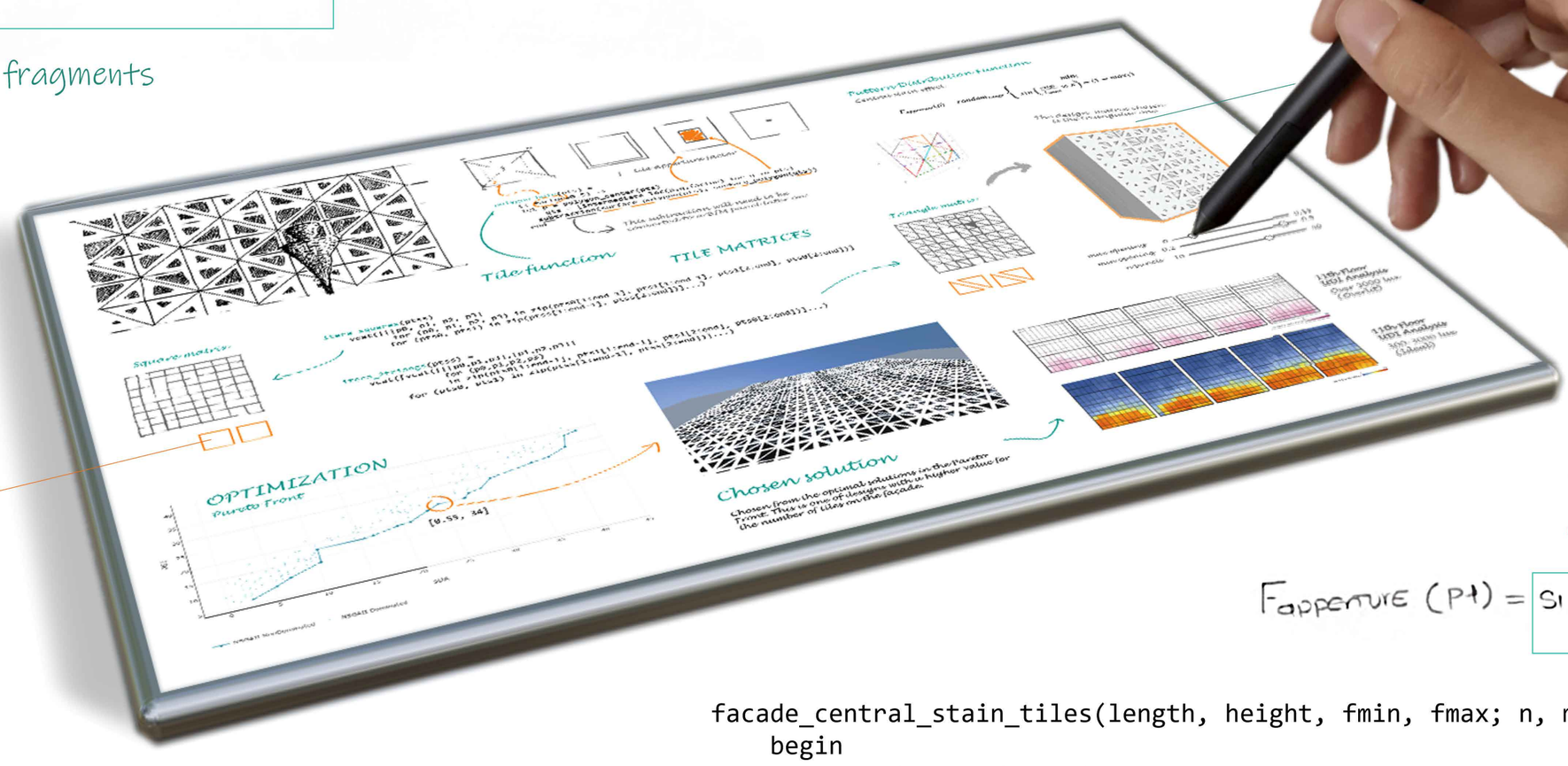
```
polygonal_tile(pts) =
    let p = polygon_center(pts)
        f = sin(p.x/length*effect)
        qts = [intermediate_loc(c, p, f) for p in pts]
      subtraction(surface_polygon(pts), surface_polygon(qts))
    end

# effect -> new parameter (controls the appertures' effect)
# qts -> intermediate locations between p and pts
```



Written annotations



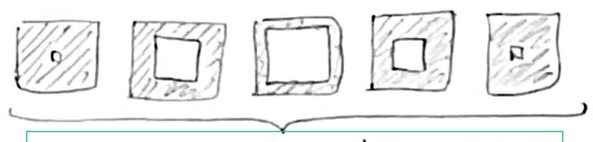
Program fragments



Mathematical design principles

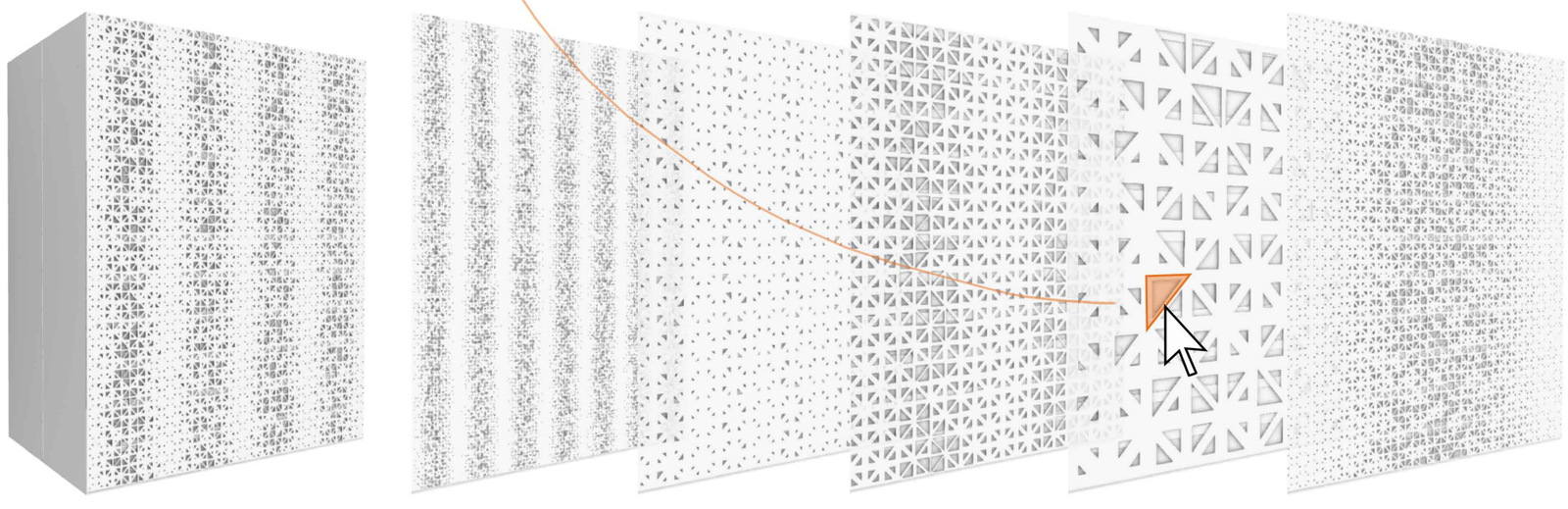
$$F_{apperture}(Pt) = \sin\left(\frac{Pt.x}{length} \times Pi\right)$$

```
facade_central_stain_tiles(length, height, fmin, fmax; n, m) =
    begin
      set_random_seed(1234)
      pattern.(polygonal_tile,
               itera_2triangs_mirrorXY(
                map_division(xy, 0, length, n, 0, height, m)),
               factor = pts -> random_range(fmin,
                                            max(fmin,
                                                sin(pts[1][1].x/length*π)-(1-fmax))))
                                  # apperture reaches its maximum in the middle
    end
```



Written annotations

appertuie reaches its maximum in the middle



Algorithm-drawing relationships
Algorithm-model relationships

TRACEABILITY