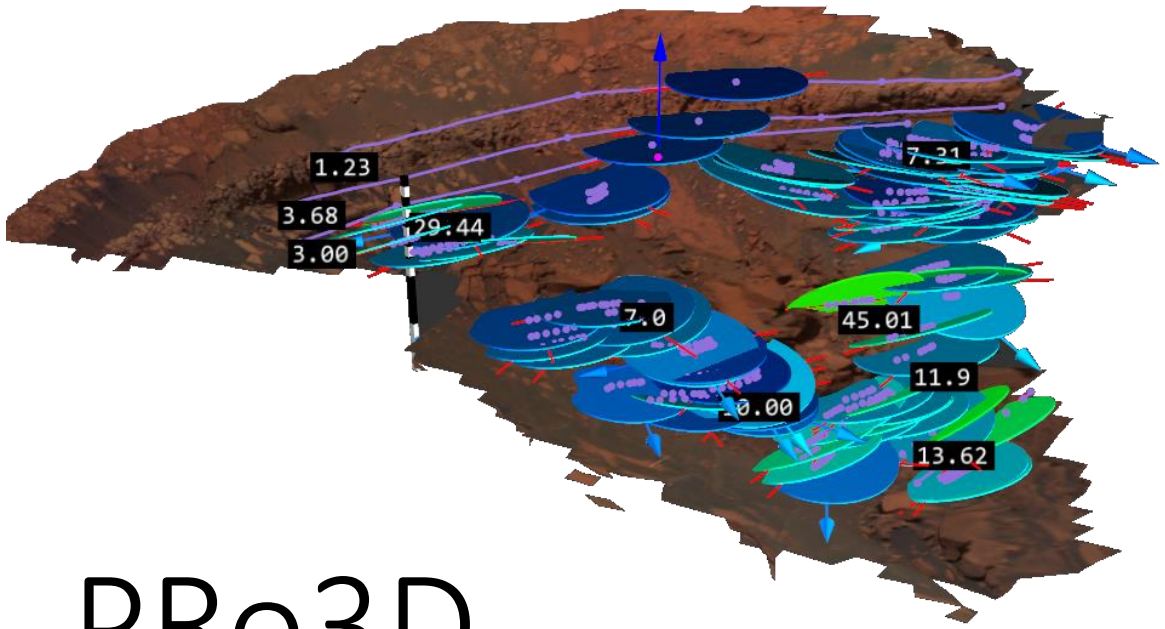
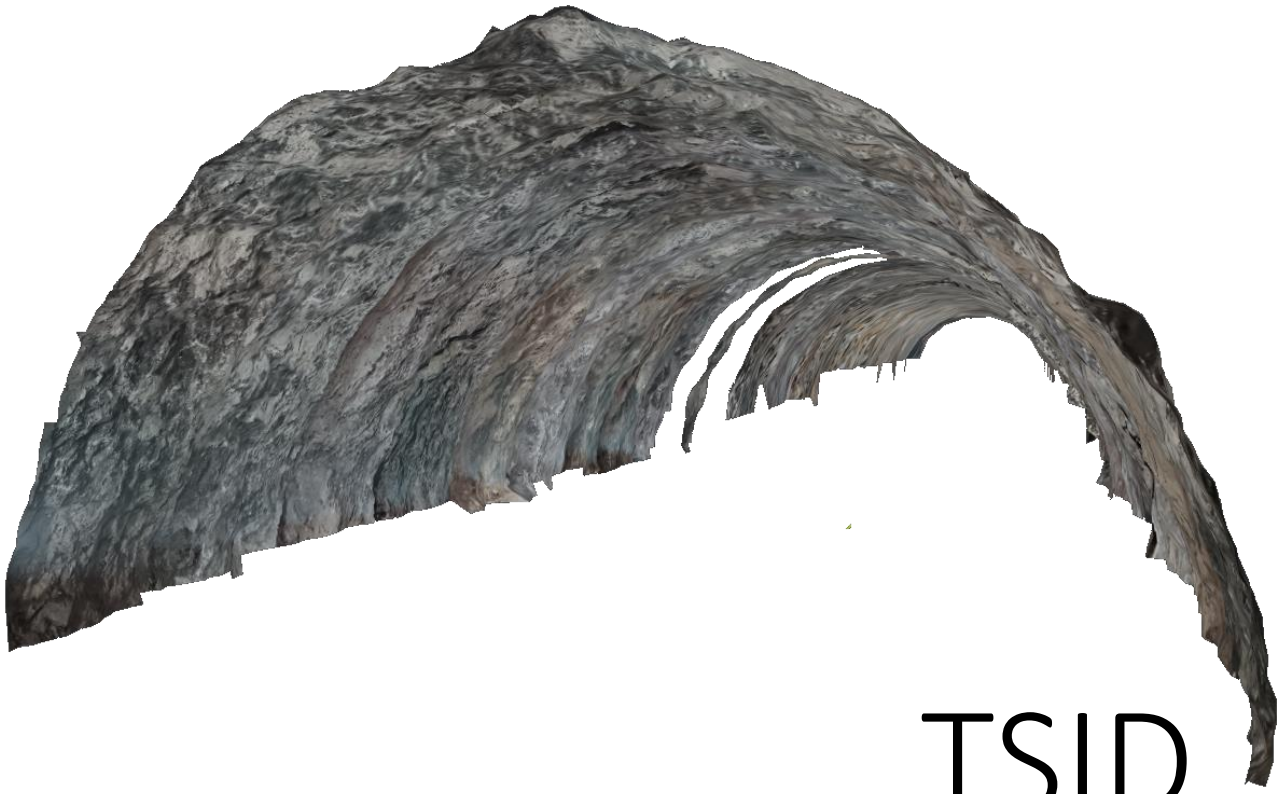


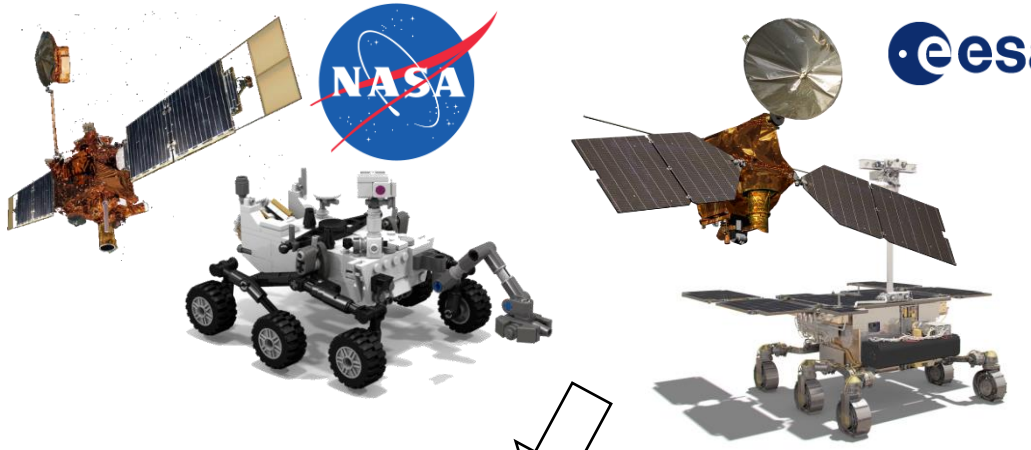
PRo3D

Planetary Robotics 3D Viewer

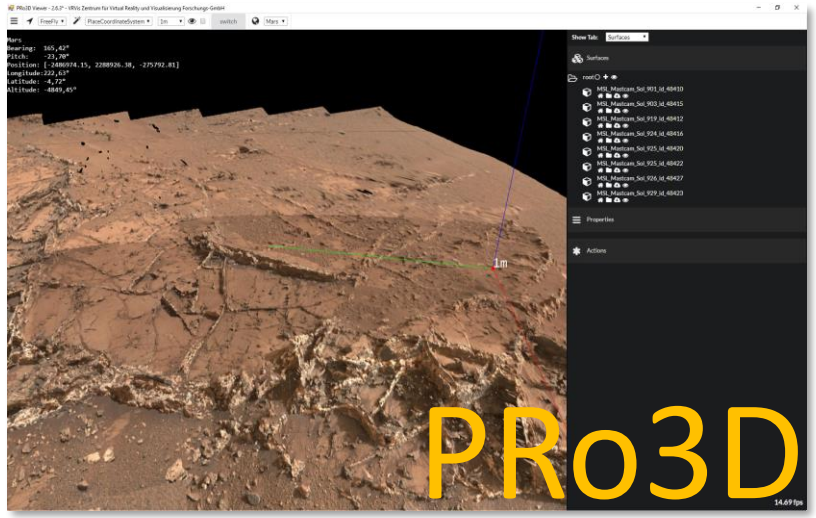


TSID

Tunnel Surface Inspection and Documentation



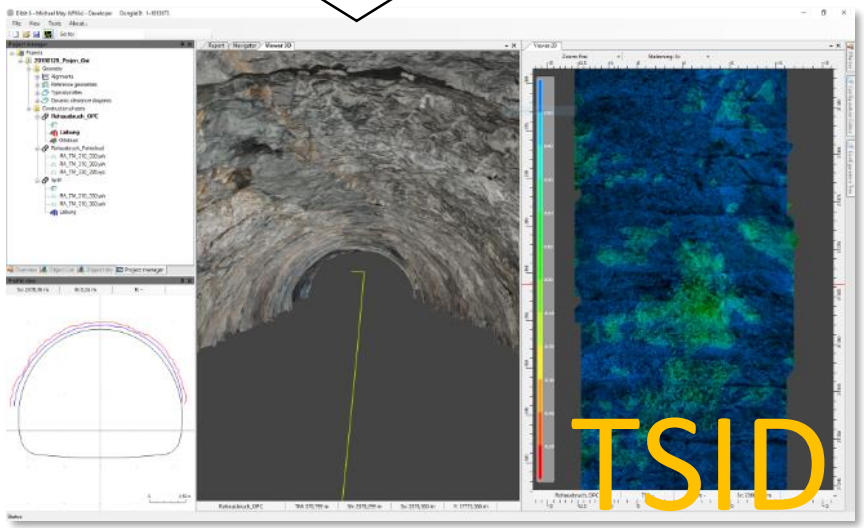
JOANNEUM RESEARCH



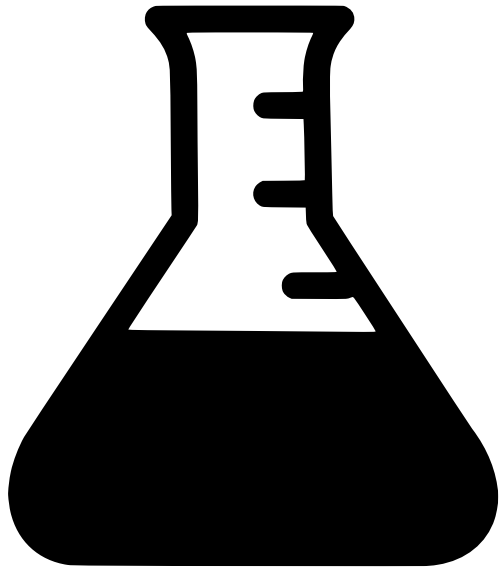
Images / Laser Scans

Reconstruction

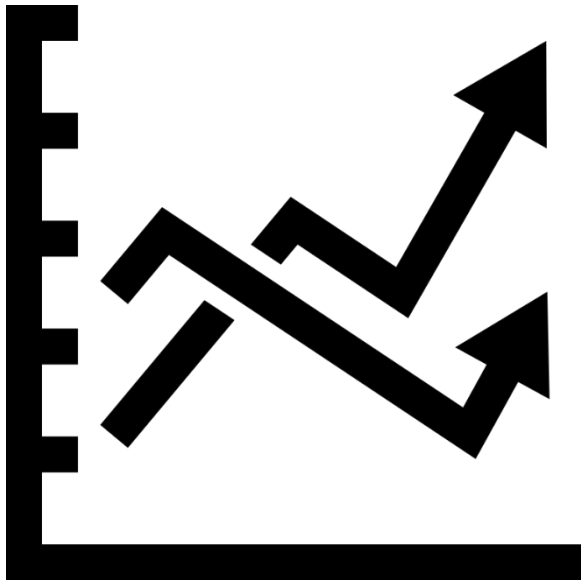
3D Surfaces



https://www.dibit.at/dienstleistungen/tunnelbau/tunnelscanning/

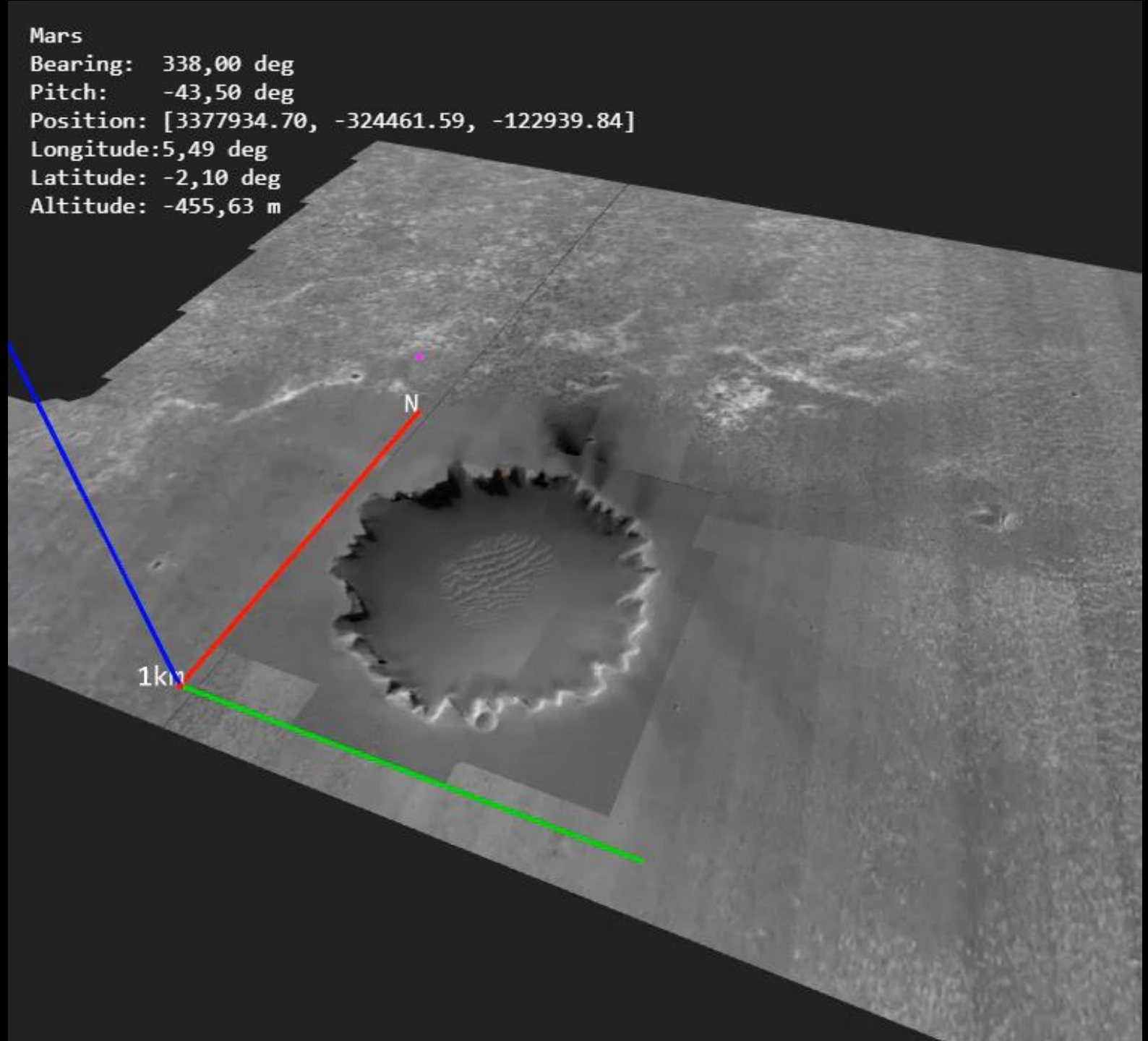


Visualization
Prototype



Short Time
to Market

Low
Budget


Mars
Bearing: 338,00 deg
Pitch: -43,50 deg
Position: [3377934.70, -324461.59, -122939.84]
Longitude:5,49 deg
Latitude: -2,10 deg
Altitude: -455,63 m
N
1km


327

MOLA
(orbiter)

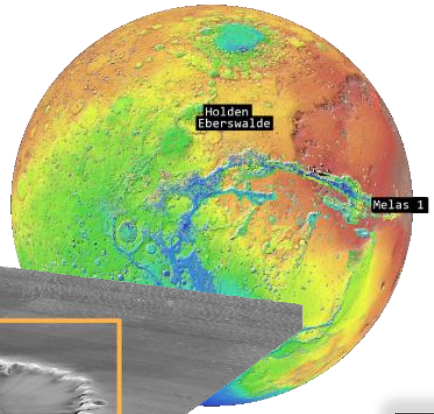
HiRISE
(orbiter)

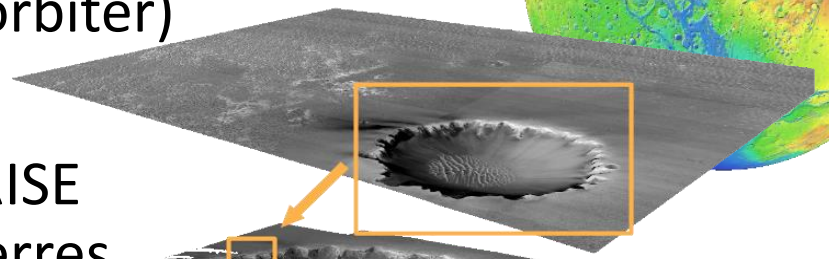
HiRISE
superres

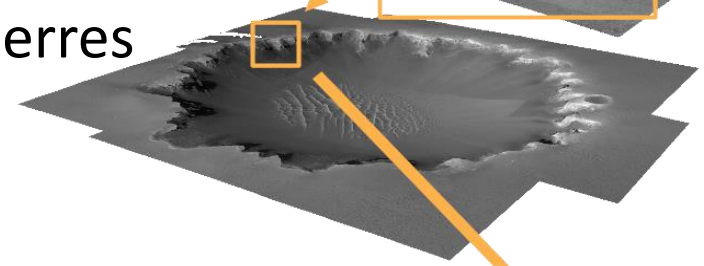
Pancam
(rover)

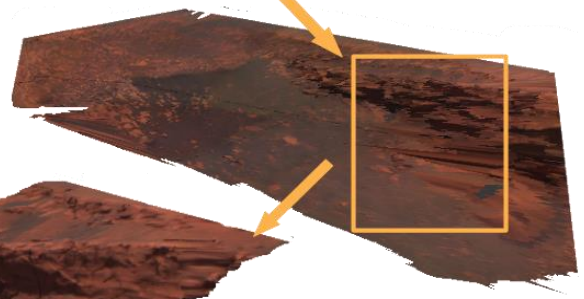
Pancam
widestereo

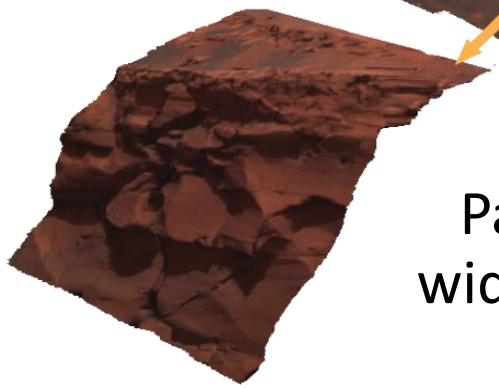
...



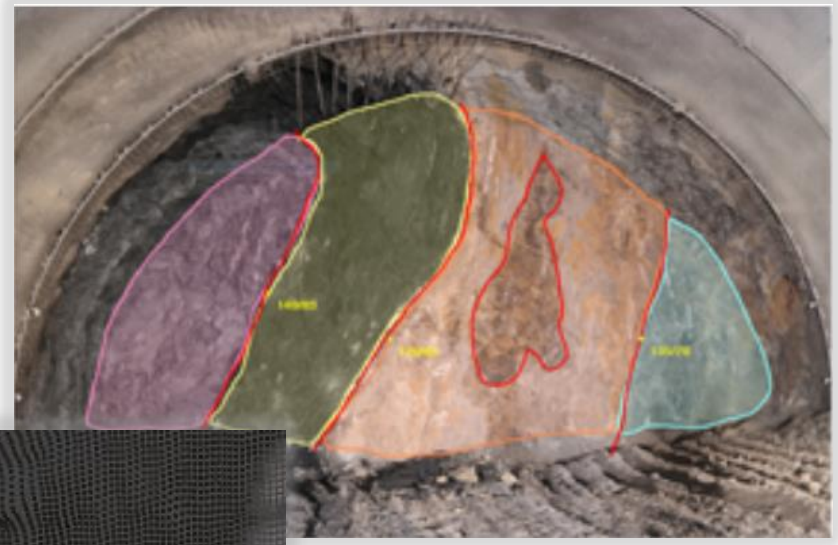


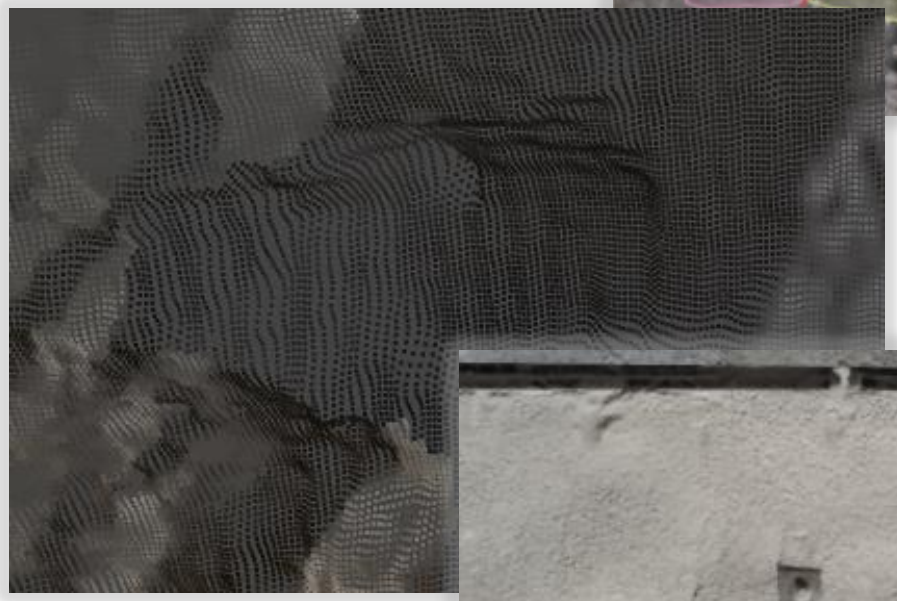






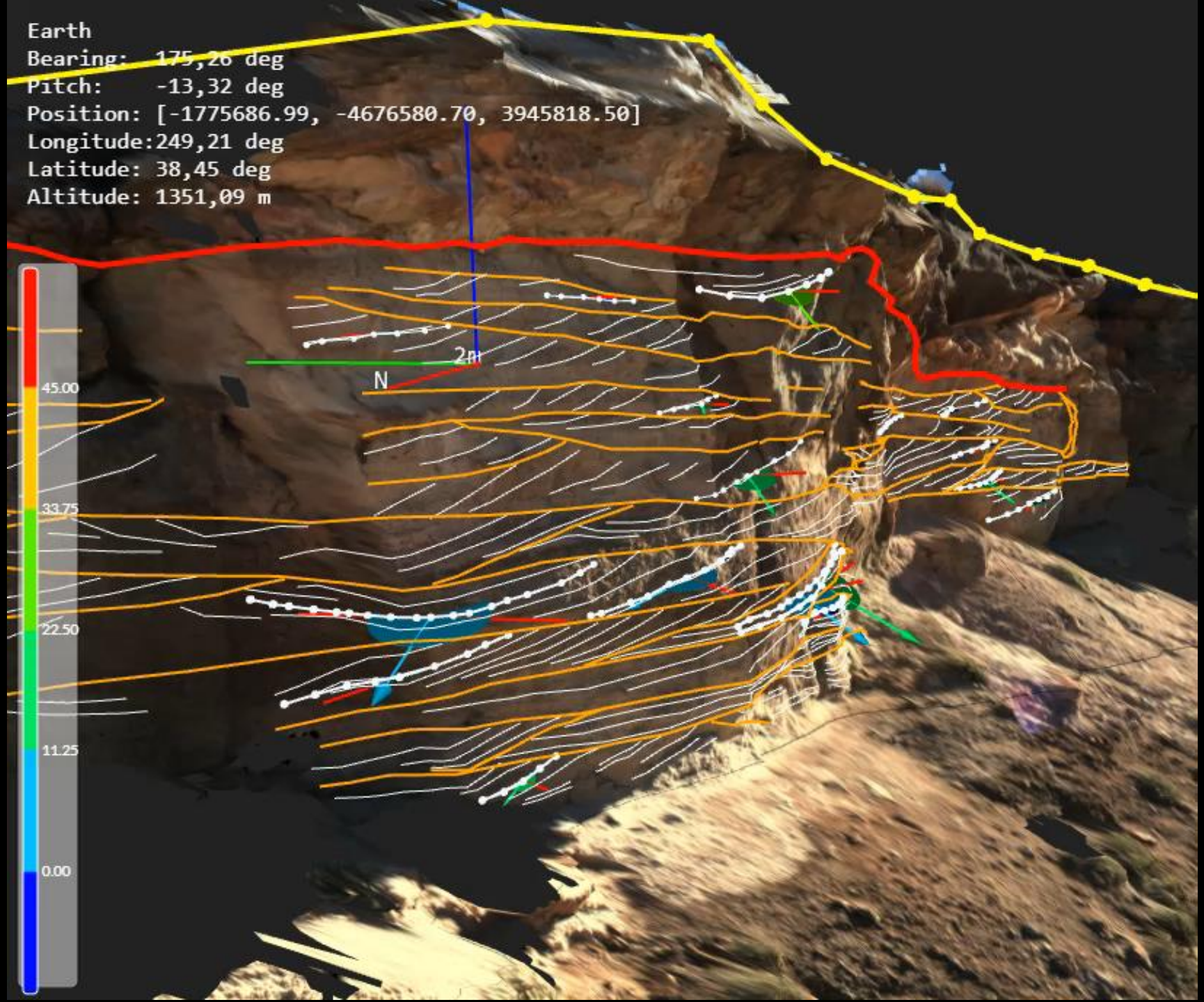
Handheld
Images





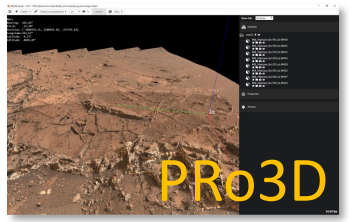
Pointclouds

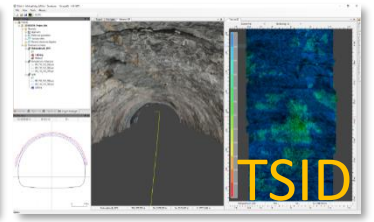
Triangles
(OPC)


Earth
Bearing: 175,26 deg
Pitch: -13,32 deg
Position: [-1775686.99, -4676580.70, 3945818.50]
Longitude:249,21 deg
Latitude: 38,45 deg
Altitude: 1351,09 m
45.00
33.75
22.50
11.25
0.00
2m
N


Presentation
Interaction
COMPLEXITY

+8 years in development





maintenance

DEADLOCK

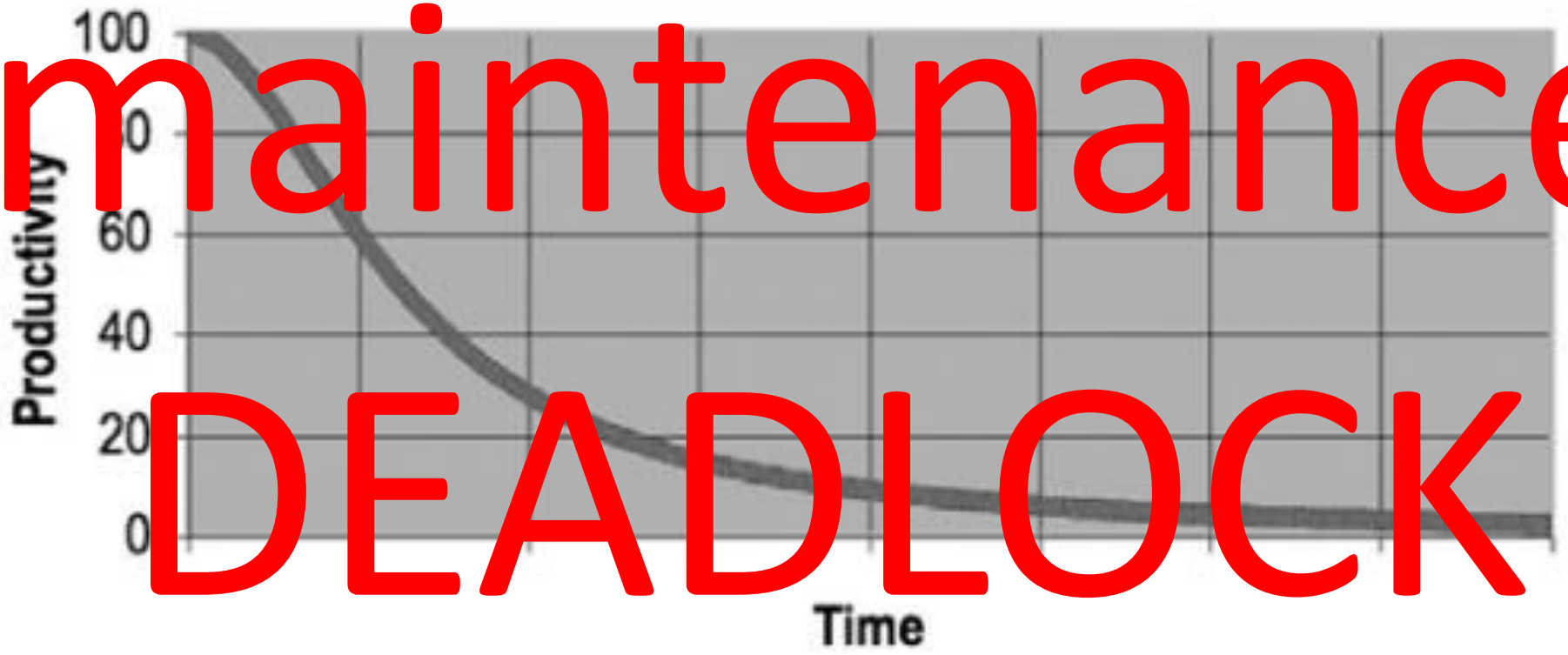


Figure 1-1

Productivity vs. time

Robert C. Martin – Clean Code

Sources of Complexity

- Doing **complex things**
- **Interconnections** between modules
- 3D / UI / logic **separation**
- Changing **scope**
- **Make it work** first optimize later

How to tame Complexity?

Can’t remove inherent complexity

Rumor has it, **Functional Programming** might be the key

- Concise
- Convenient
- Correctness
- Concurrency
- Composability*

https://fsharpforfunandprofit.com/why-use-fsharp/

https://live.staticflickr.com/2307/32063430274_bf8da24b81_b.jpg

Functional Rewrite!

- Two applications – 8+ years in development
- Moving from C# OOP WPF to F# FP ?GUI?
- Functional paradigm of **Immutable Data**

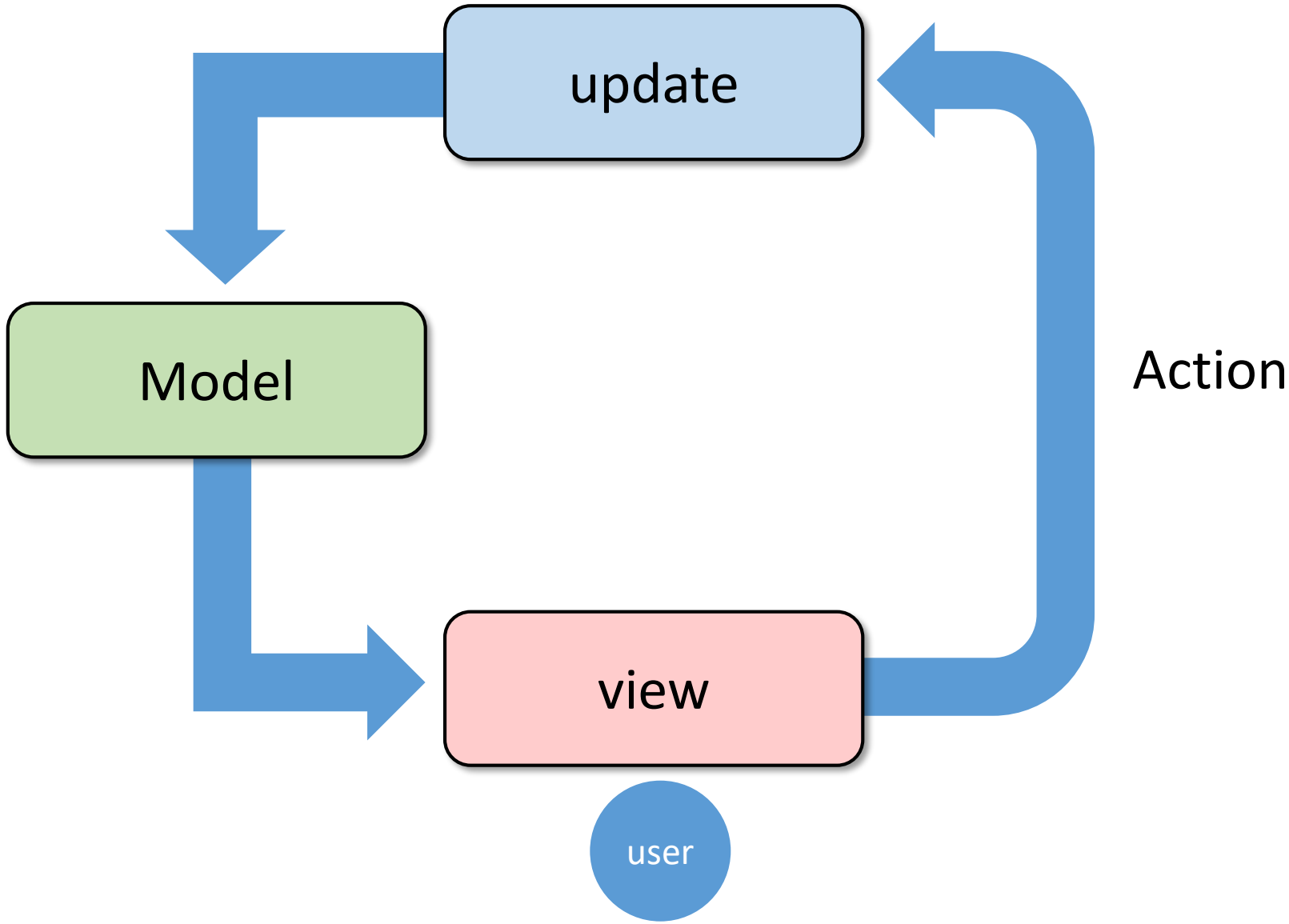
```
//mutable
public static void Add(Dictionary<string,int> d, string key, int value)

//immutable
public static Map<string,int> Add(Map<string,int> m, string key, int value)
```

Immutable data feasible for a **whole Application**?

```
public static Scene AddObject(Scene m, string filepath)
```

```
public static Scene ChangeCamera(Scene m, Matrix44f view)
```

ELM Architecture



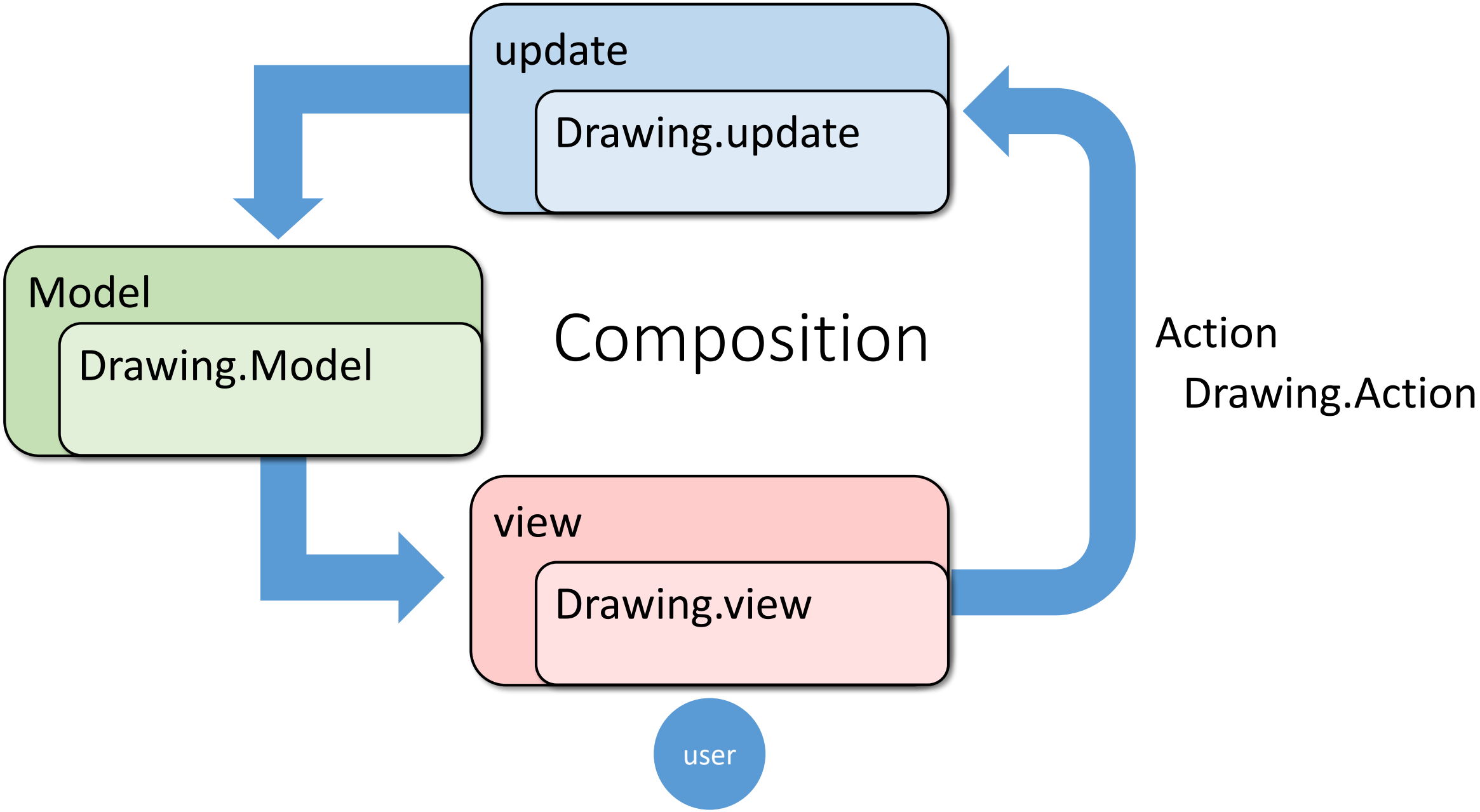
```
type Polygon = { points : list<V2d> }

type Model =
{
      polygon : Polygon
      cursor  : option<V2d>
}

type Message =
   | AddPoint of V2d
   | MoveCursor of V2d

let update (m : Model) (msg : Message) =
  match msg with
    | AddPoint pt ->
       { m with polygon = { points = pt :: p.points } }
    | MoveCursor v ->
       { m with cursor = Some v } // set the current cursor
```

```
let view (m : Model) =
    let viewPolygon points =
      points |> pairwise |> List.map (
        fun (p0,p1) -> line p0 p1 [style "stroke:rgb(0,0,0);"])

    body [] [
          button [onClick Undo] [text "Undo"]
          span [] []
          button [onClick Redo] [text "Redo"]
          br []
          viewPolygon m.polygon.points
          br []
      ]
```

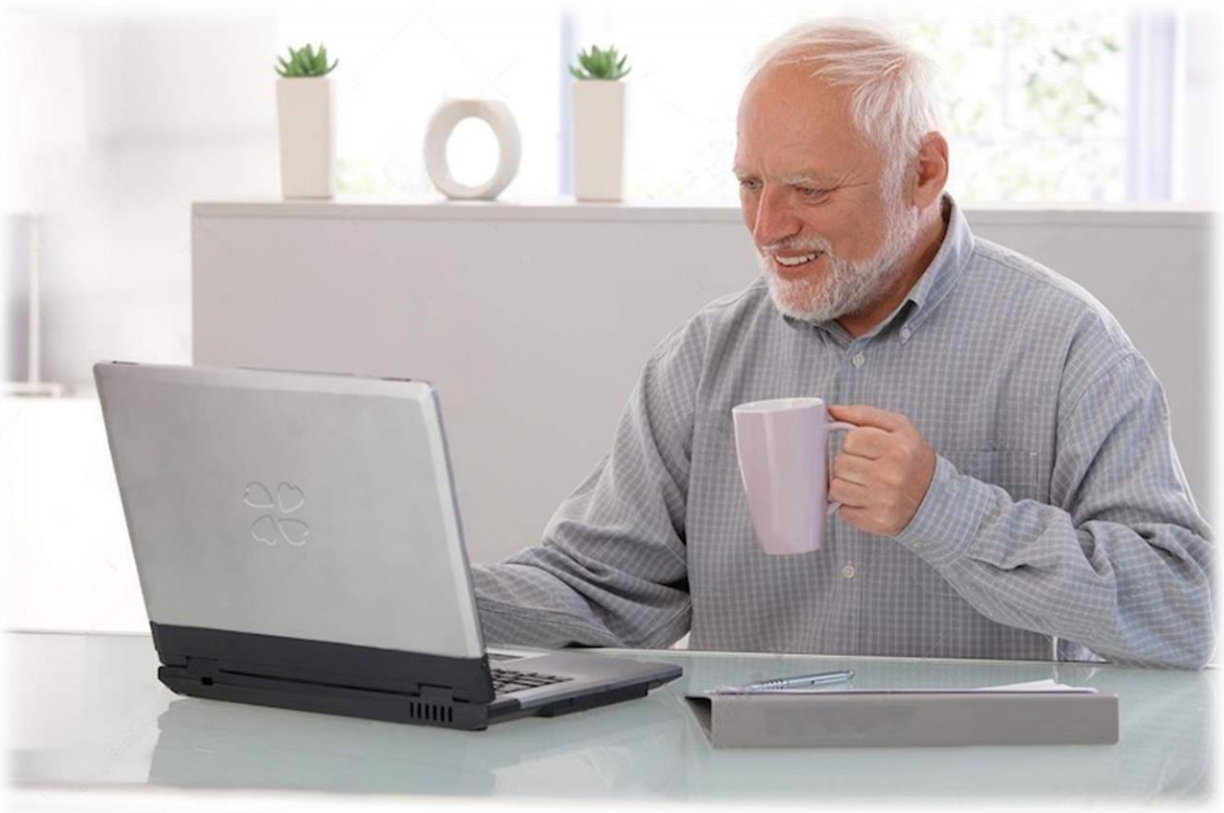
```
type Model =
{
      polygon : option<Polygon>
      cursor  : option<V2d>
      past    : option<Model>
      future  : option<Model>
}
```

```
type Message =
   | AddPoint of V2d
   | MoveCursor of V2d
   | Undo
   | Redo
```

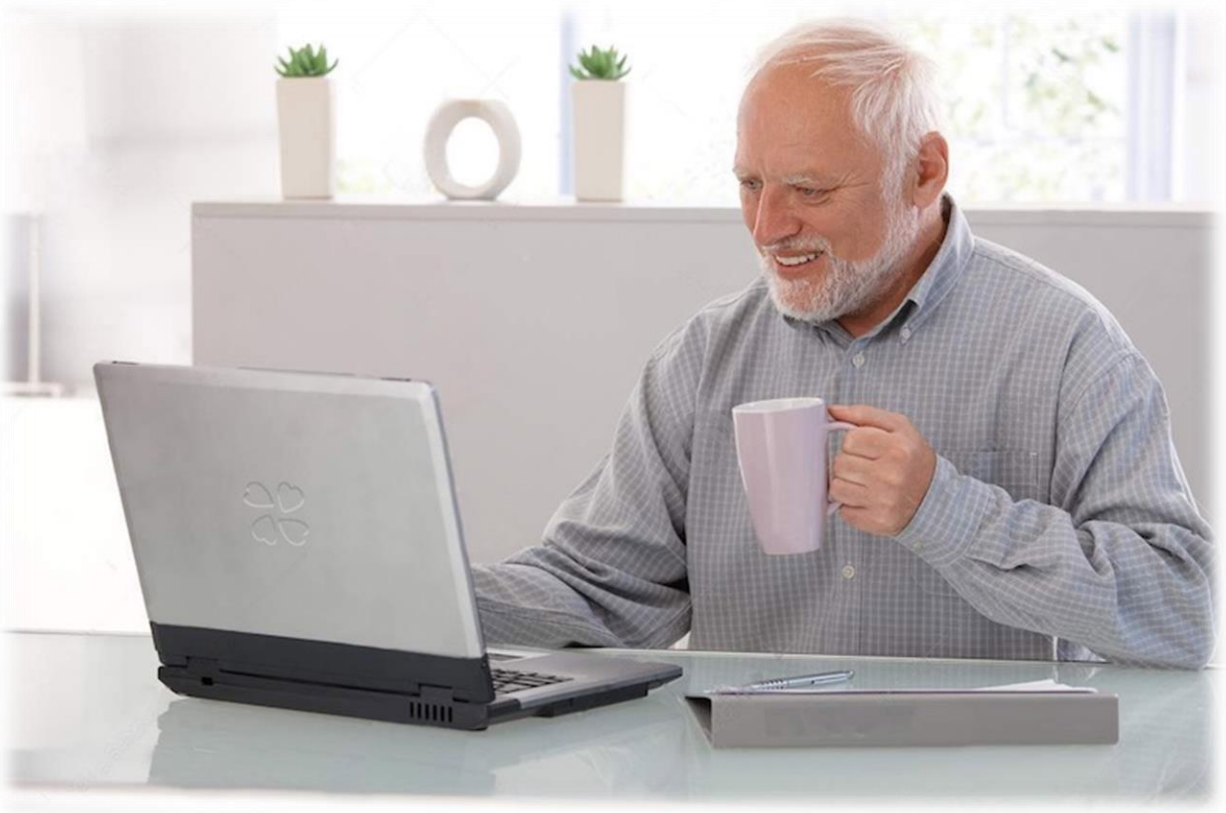
```
let update (m : Model) (msg : Message) =
   match msg with
    | AddPoint pt ->
       { m with polygon = { points = pt :: p.points}; past = Some m }
    | Undo _ ->
       match m.past with
        | None -> m // no past => nothing to undo
        | Some p -> { p with future = Some m }
          // puts the current model into the future of the new model
```

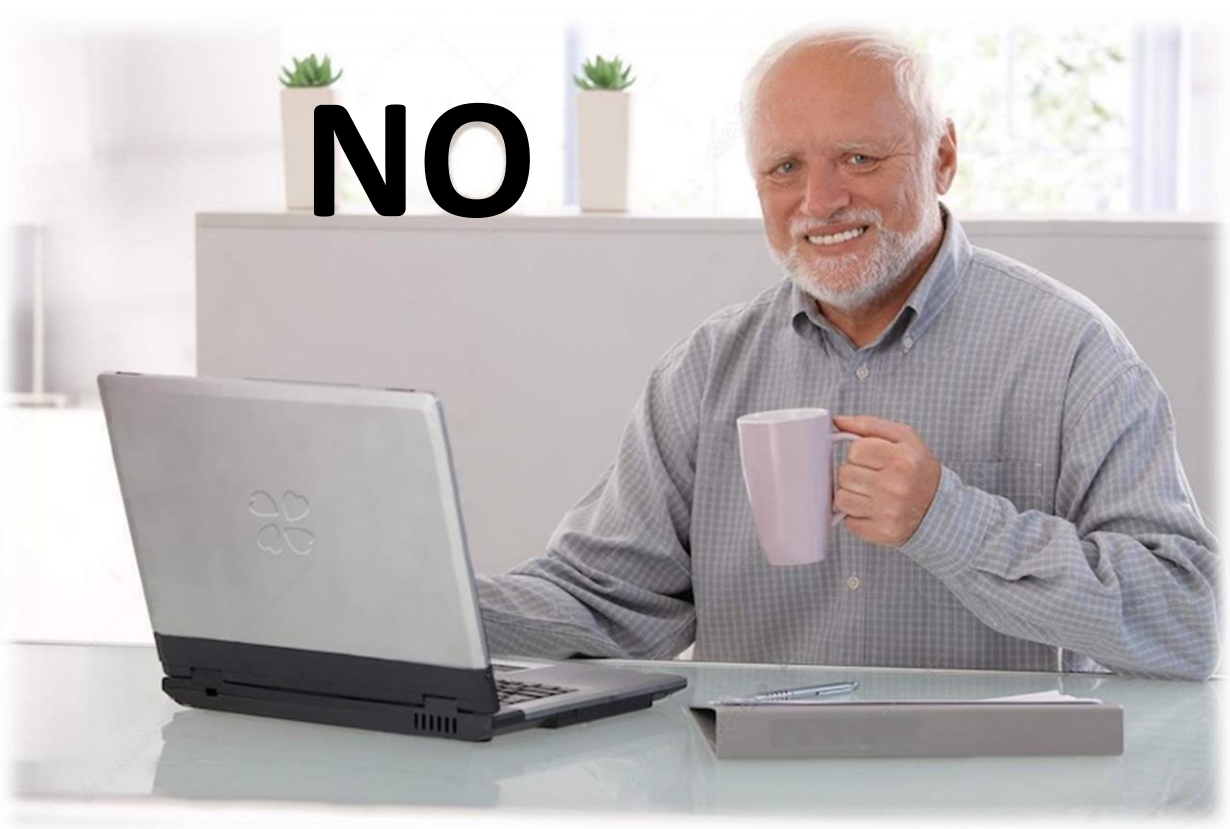

update
Drawing.update
Model
Drawing.Model
Composition
Action
Drawing.Action
view
Drawing.view
user

```
type Model = {
   scene : Scene
   drawing : Drawing.Model
   // ..about 20 other things

   past   : option<Model>
   future : option<Model>
}

let update (surf:Surface) (m:Model) (msg:Message) =
  match msg with
  | DrawAnnotation inner ->
    let drawing = Drawing.update m.drawing inner
    { m with drawing = drawing }

// … a lot of stuff
```



Have we tamed complexity?





What about efficiency?

Performance and Efficiency

How can we deal with expensive visualization functions?

For 3D graphics: Functional programming vs high-performance computer graphics, GPU Day 2018

Revisit of our 5Cs

- **Concise**
  less coding noise, fewer LOC, ‘reasonable’ code
- **Convenient**
  type system, pattern matching, higher order functions
- **Correctness**
  compile time errors instead of runtime error,
  no null, no side effects
- **Concurrency**
  event handling, sharing immutable states
- **Composability**
  maintainable, testable, reusable modules

Take Home

(1) Don’t fear the rewrite, it pays of shortly

(2) Team

- Increased motivation
- Steep learning curve (esp. for OOP trained)

(3) Don’t throw away and port tested code (rather wrap)

Take Home

(4) FP fits high performance applications (if done right)

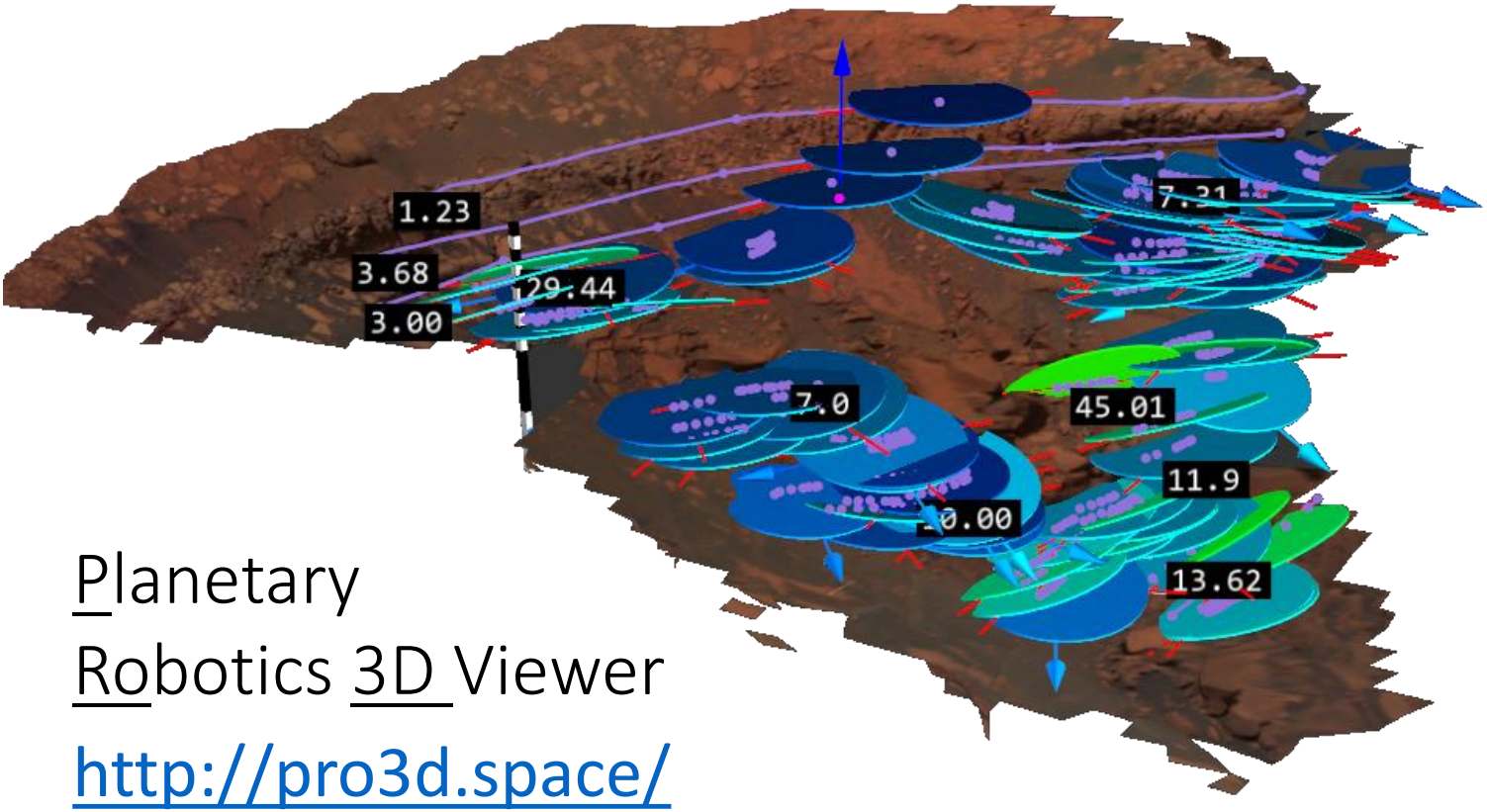
- Use diffing algorithm to translate immutable snapshots to efficient GPU updates

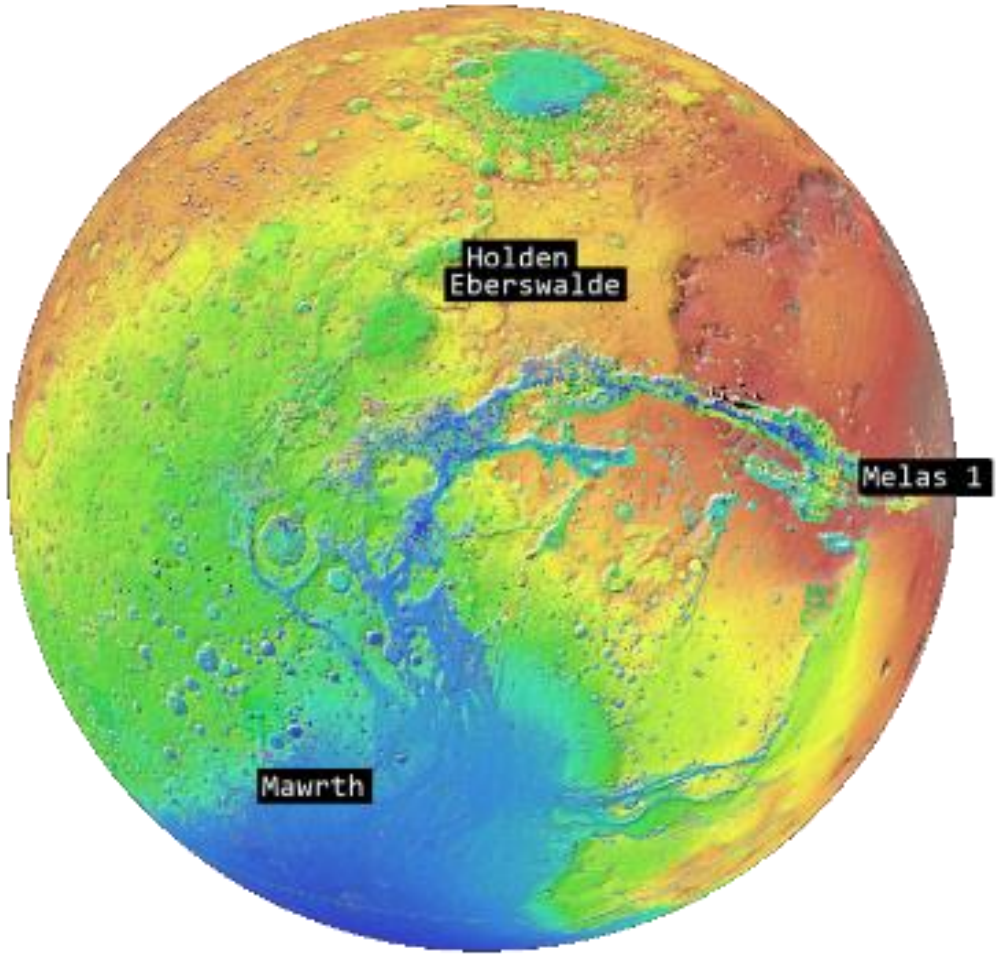
(5) F# plays well with others

- .NET runtime / C++ marshalling
- GPU / compute shader

(6) Functional Programming can tame complexity !!!

Live Demo PRo3D





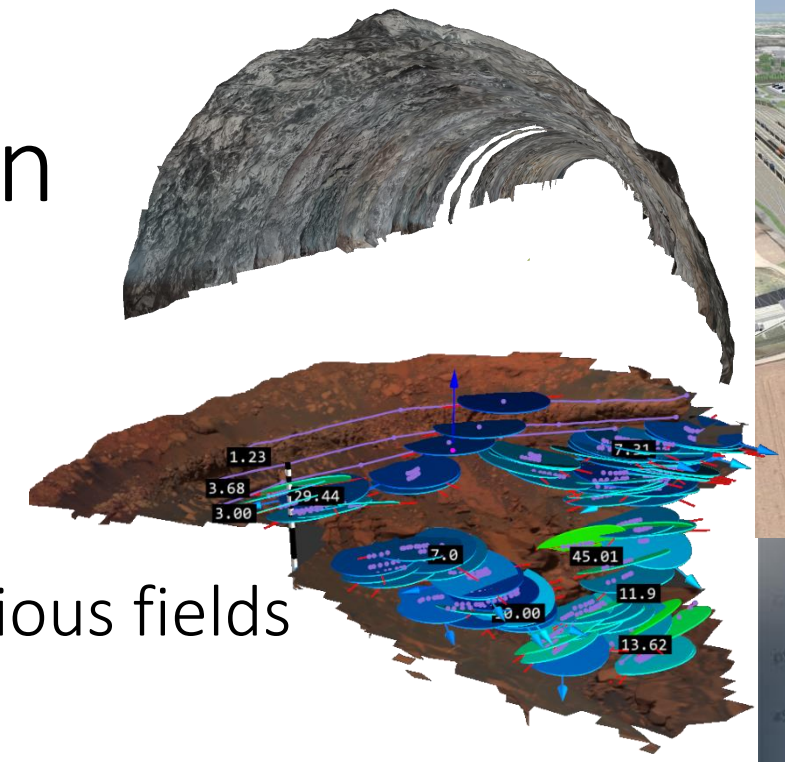
Planetary
Robotics 3D Viewer

http://pro3d.space/

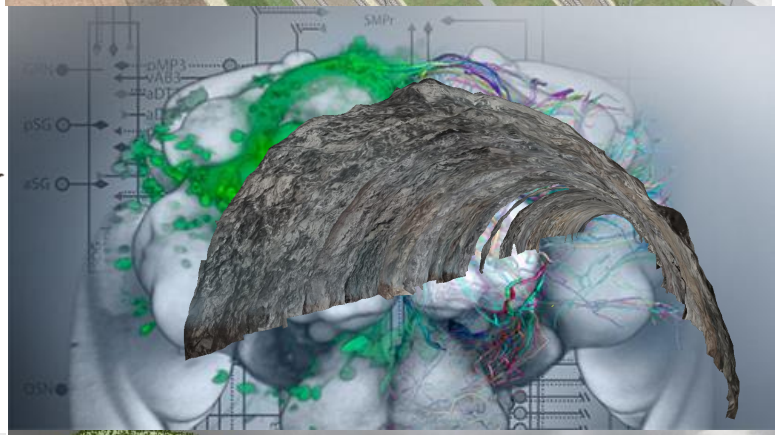
Further Reading

- “Elm: Concurrent FRP for functional GUIs”, Phd thesis 2012, Evan Czaplick https://elm-lang.org/assets/papers/concurrent-frp.pdf
- Elm diffing algorithm, https://github.com/elm/virtual-dom
- Aardvark platform, https://aardvark.graphics
  - Aardvark’s high-performance ELM implementation https://github.com/aardvark-platform/aardvark.media
- Functional programming in the wild GPU Day 2018
- Functional programming vs high-performance computer graphics, GPU Day 2018
- Domain driven design https://fsharpforfunandprofit.com/ddd/
- PRo3D project page, http://pro3d.space/
  - Will be open sourced soon...

Call for collaboration

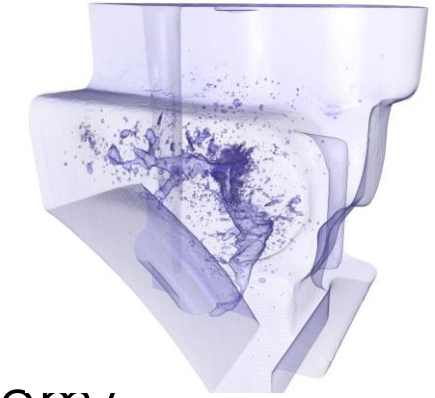


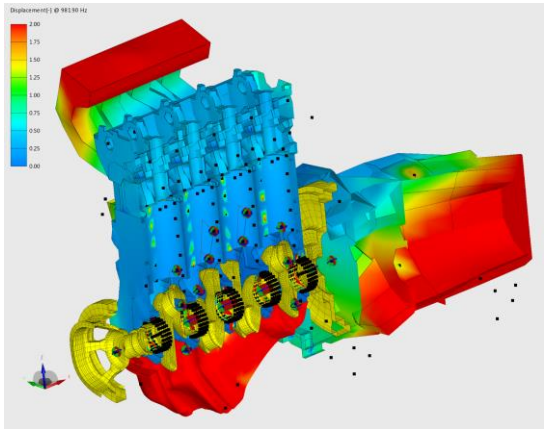
- VRVis Research Center
- Visualization research in various fields
- https://www.vrvis.at/

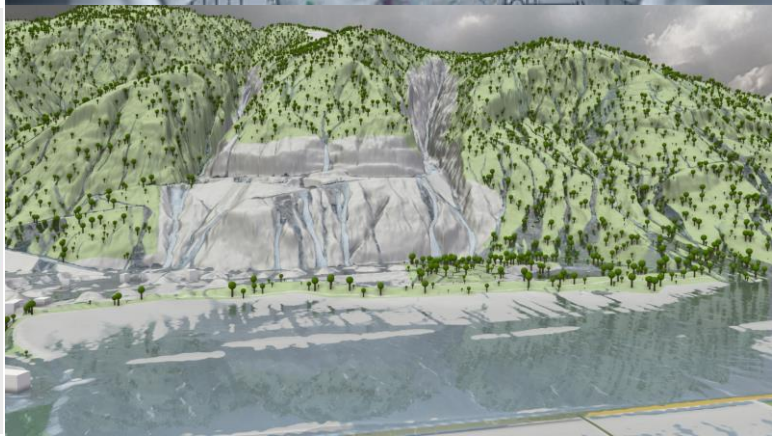


aardworx

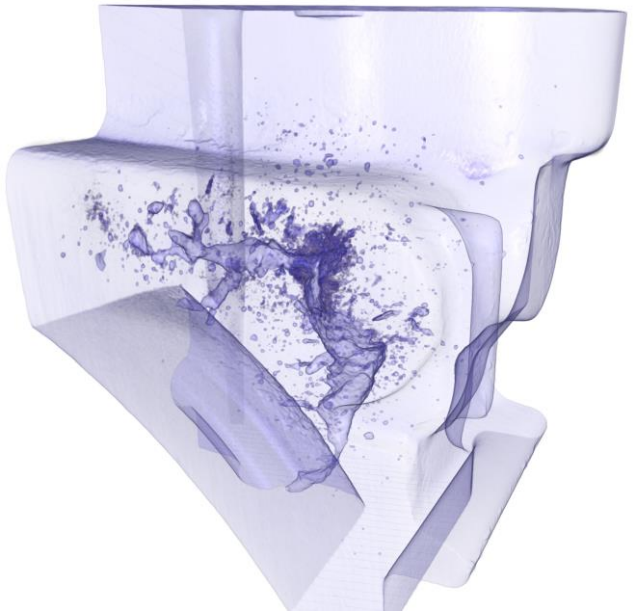
- aardworx
- Commercial HPG
- https://aardworx.com/







explore, measure, analyze 3D data in 3D



Remote (volume) rendering
cloud services

aardworx

- products and services from research
- Functional programming consulting & advice

Big (laser scan) points in browser

Solving the performance problem

- Computing difference is the key!
- Elm, React uses this approach
- For 3D graphics: Functional programming vs high-performance computer graphics, GPU Day 2018
- Ongoing work: scientific paper on this topic

Functional rewrite timeline