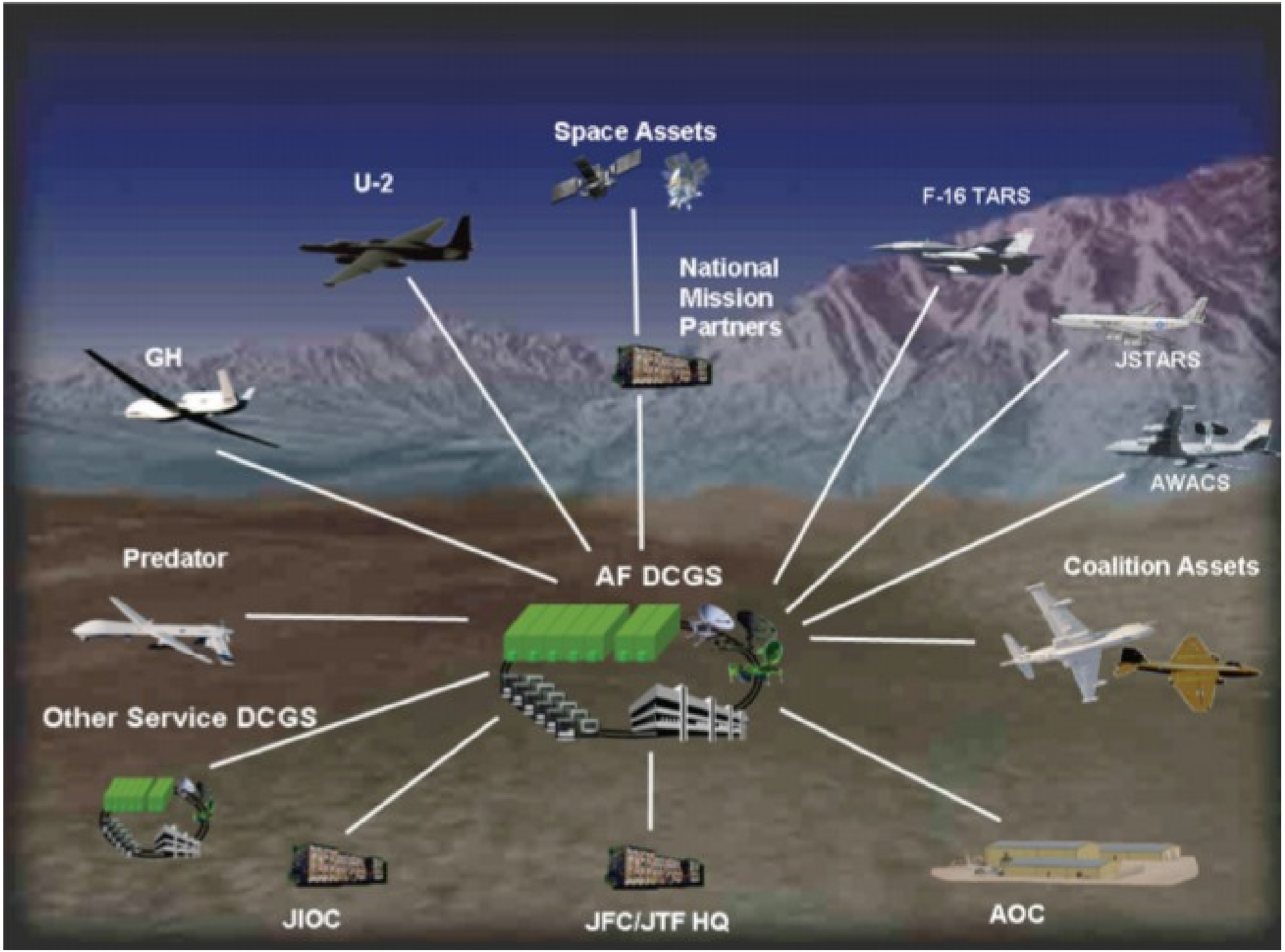
AFWERX CHALLENGE

Advanced Microelectronics Design and Prototype Challenge

THOMAS CATALANO

@tomsnode

Enhanced Ground Moving Target Indicator (GMTI) with added forensic analysis on a 14nm SoC.


Space Assets
U-2
F-16 TARS
National
Mission
Partners
GH
JSTARS
AWACS
Predator
AF DCGS
Coalition Assets
Other Service DCGS
JIOC
JFC/JTF HQ
AOC

2.4 Role of the Simulations in the Overall MAJEX07 Architecture

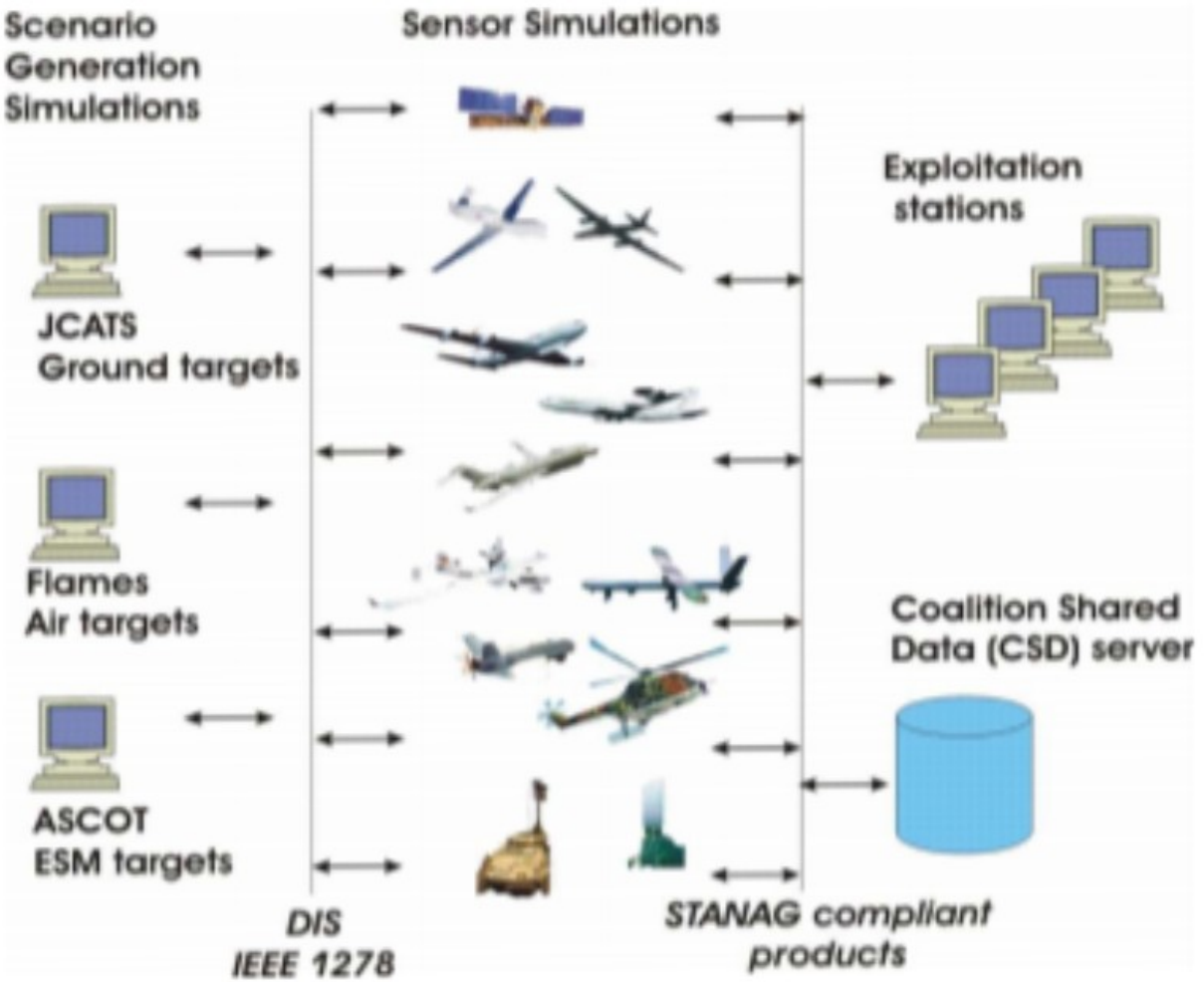


Figure 1: MAJIIC architecture

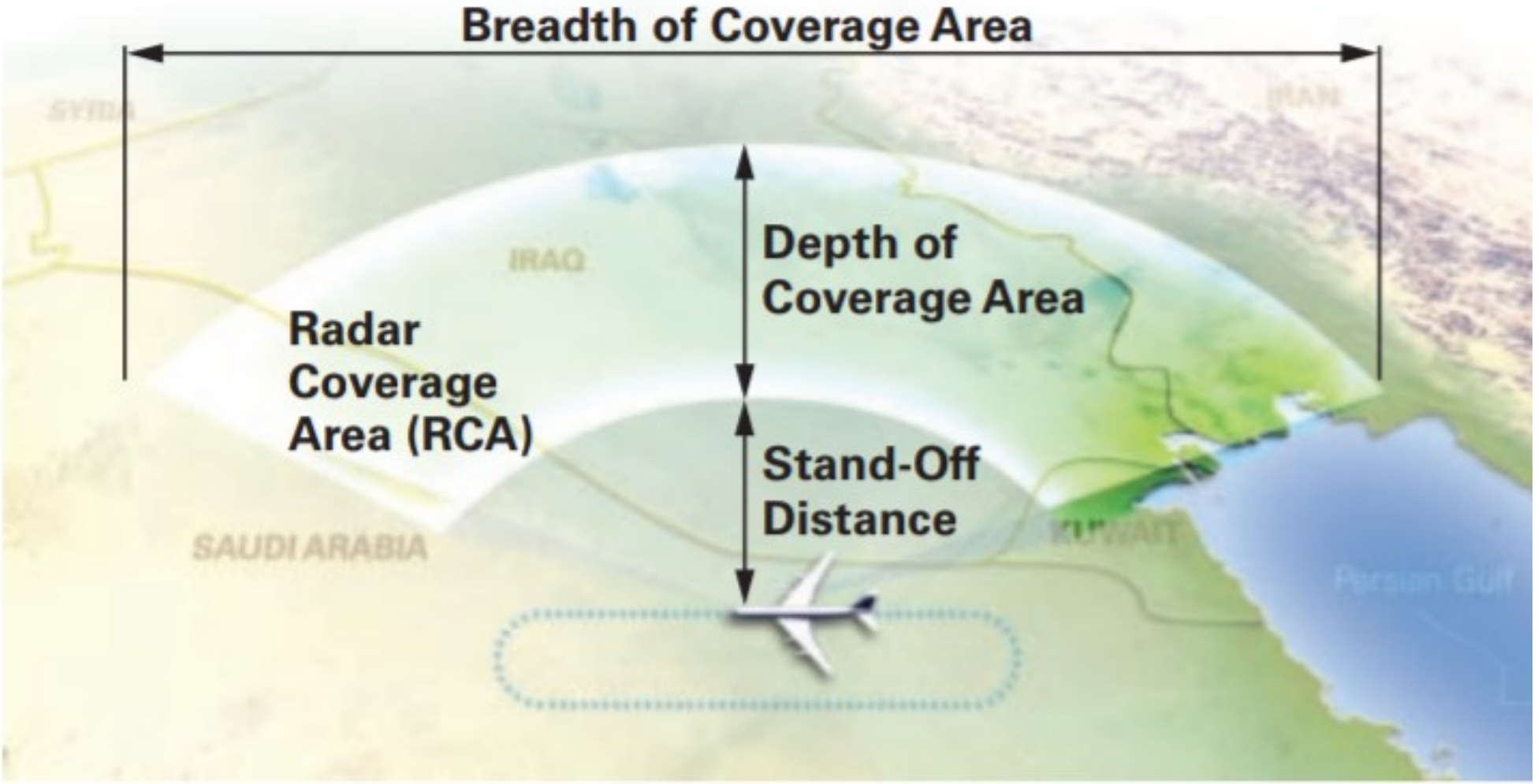
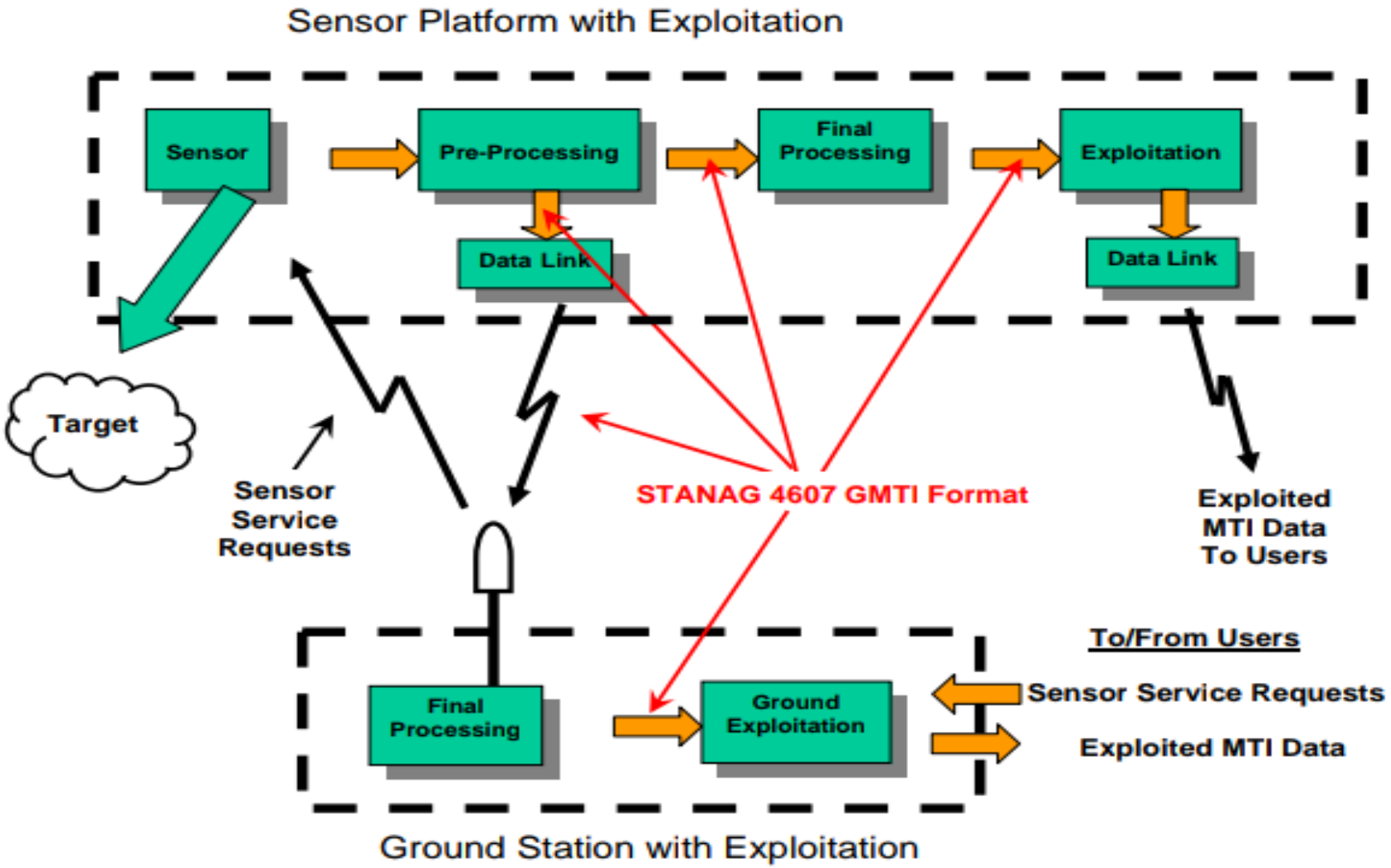


Figure A-2. Stand-off Distance and Radar Coverage Area


Sensor Platform with Exploitation
Sensor
Pre-Processing
Final Processing
Exploitation
Data Link
Data Link
Target
Sensor Service Requests
STANAG 4607 GMTI Format
Exploited MTI Data To Users
To/From Users
Sensor Service Requests
Exploited MTI Data
Final Processing
Ground Exploitation
Ground Station with Exploitation

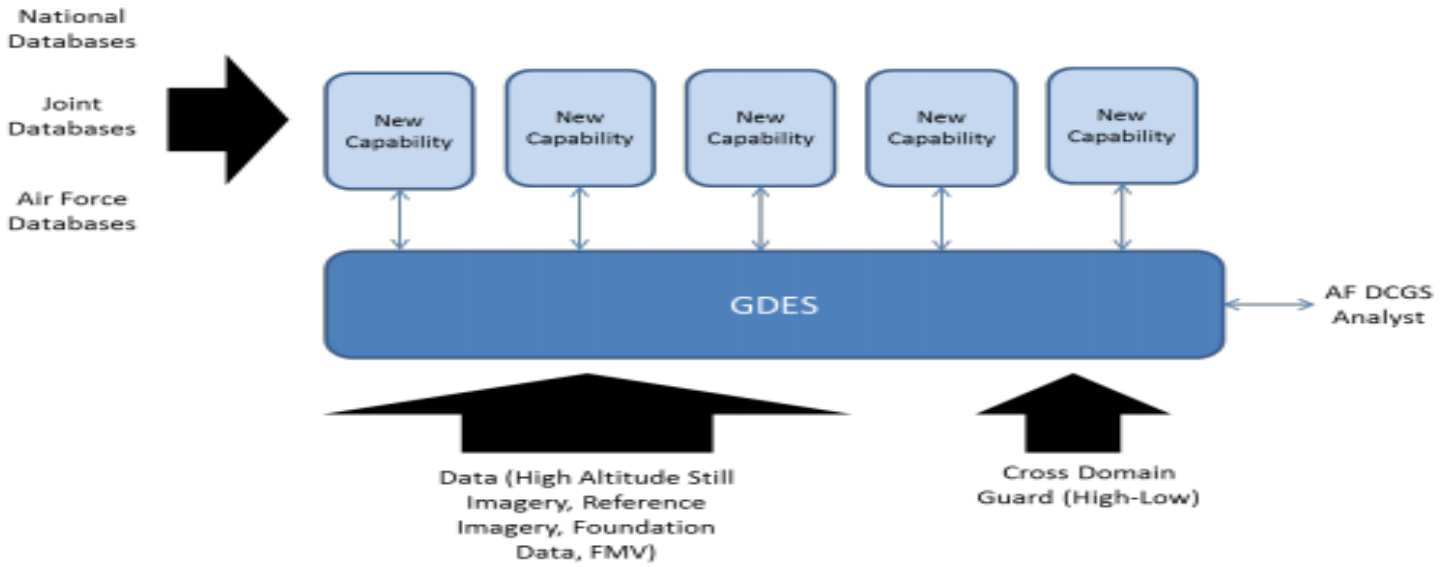
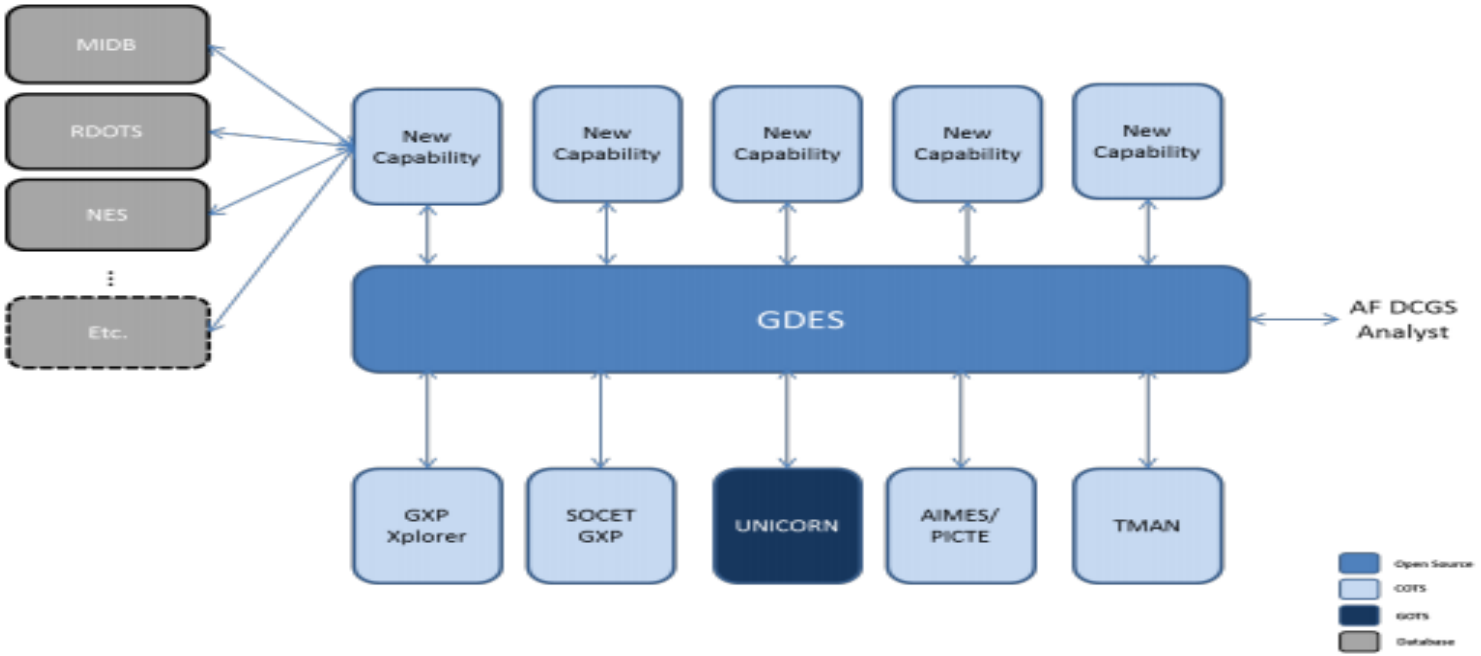


Figure 1. Proposed data flow for initiative



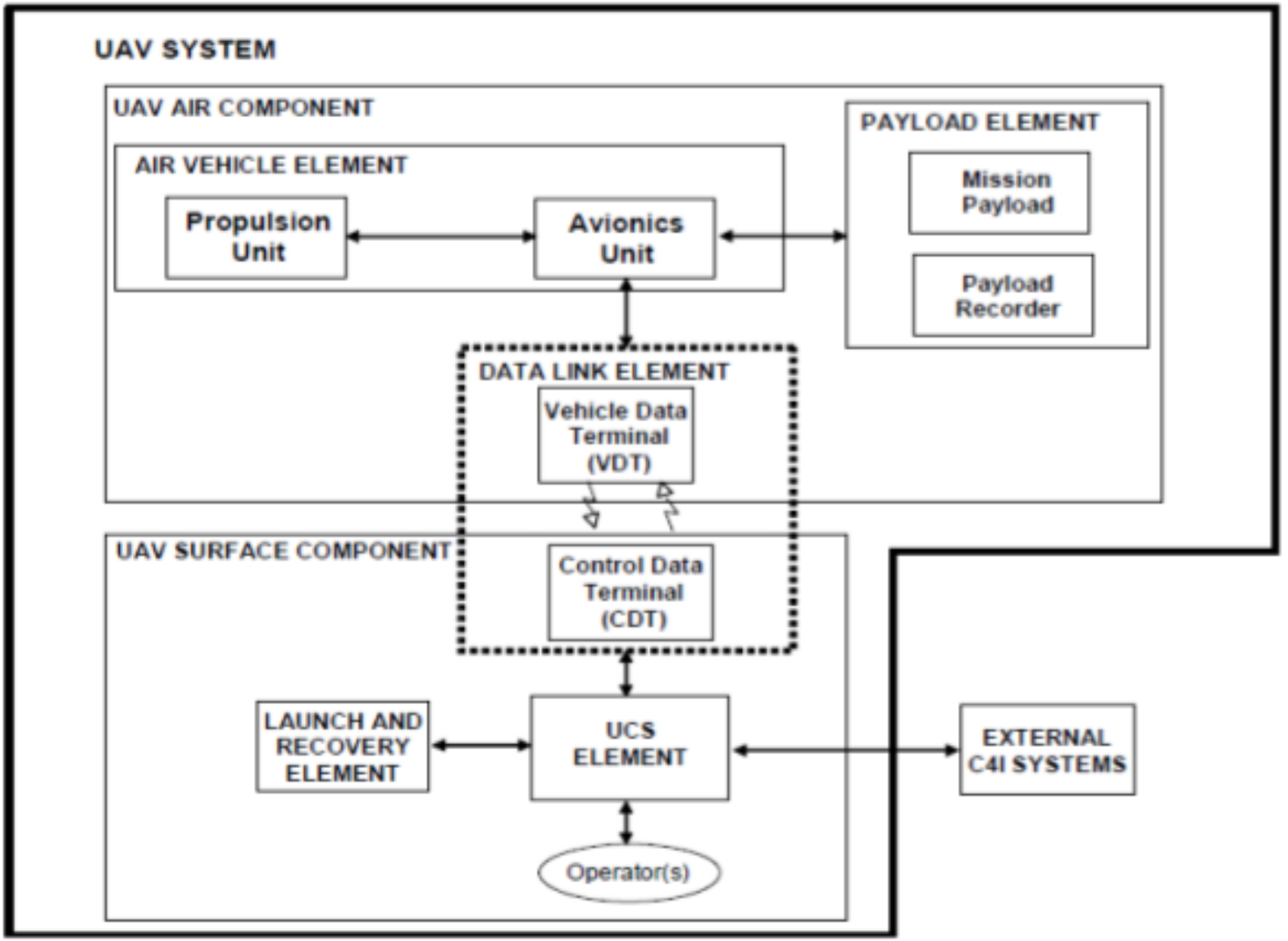


Figure 2: UAV System Elements [4].

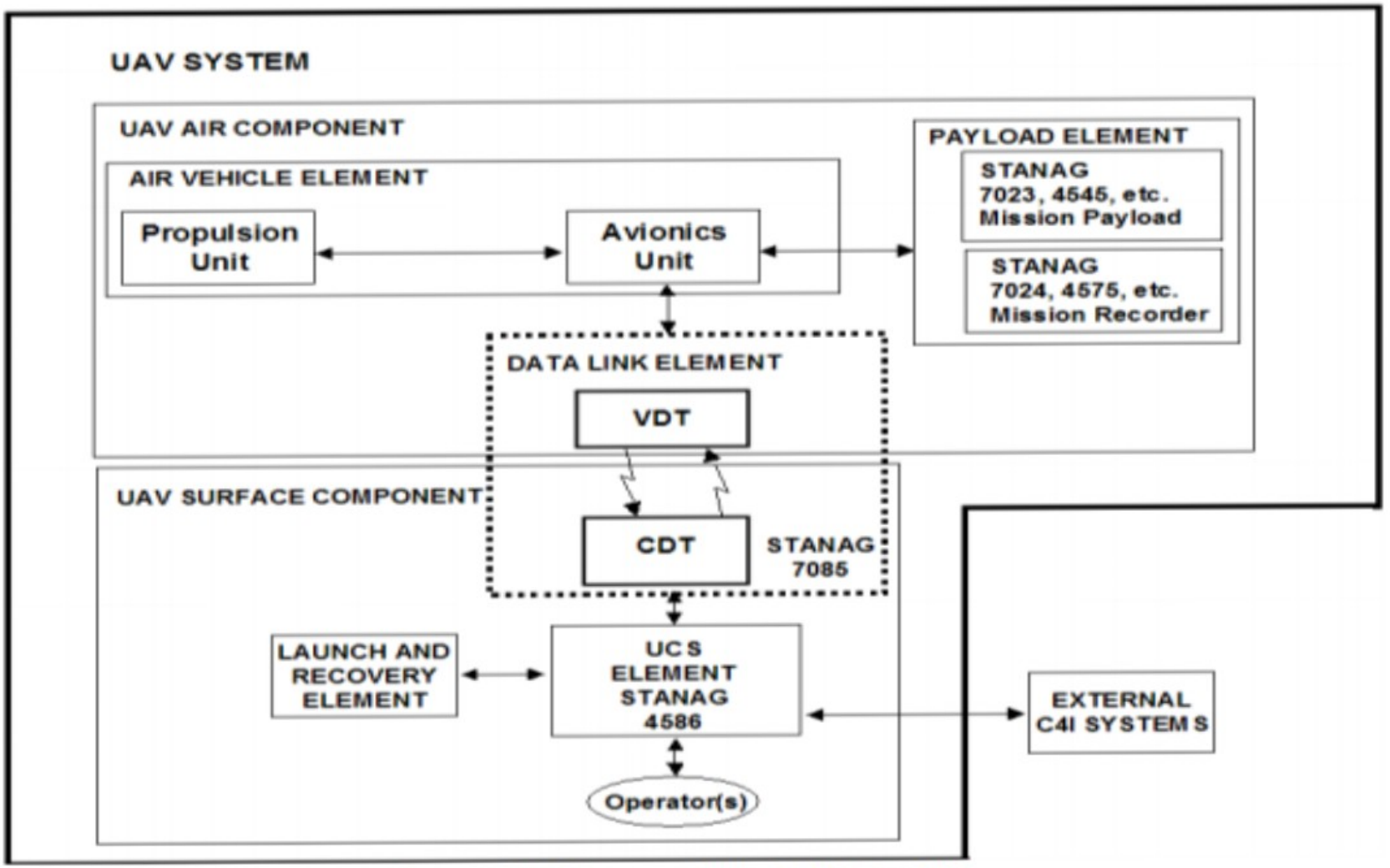
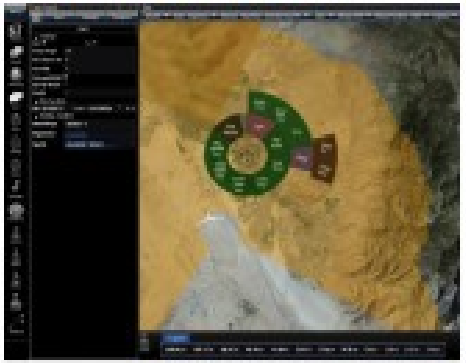


Figure 4: UAV Interoperability Architecture [4].

CLAW BENEFITS

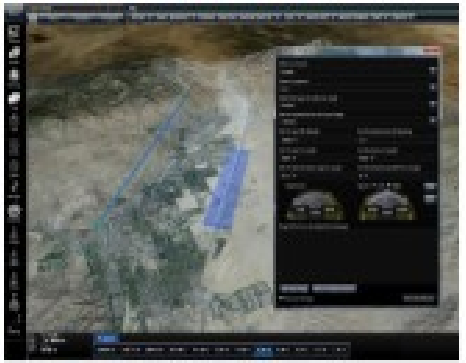
Comprehensive Dynamic Situational Awareness

ADVANCED MISSION PLANNING



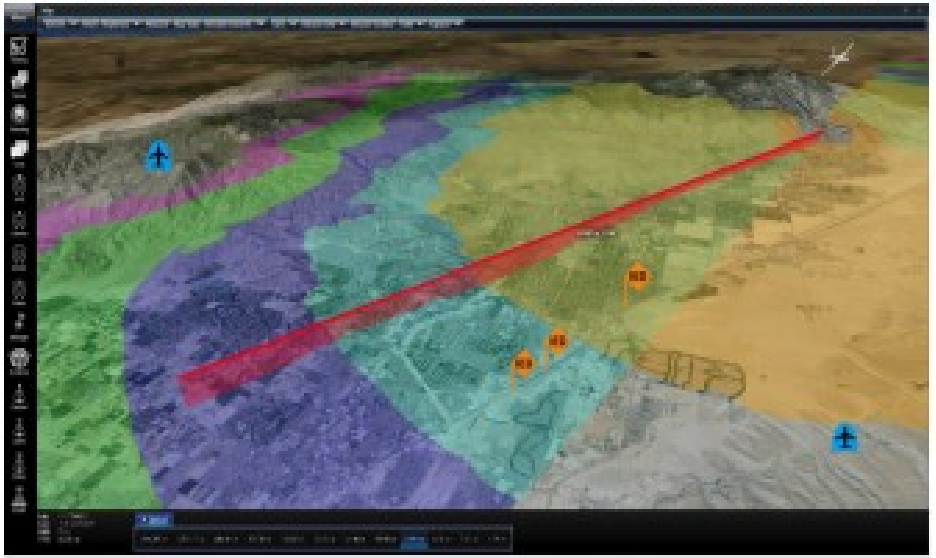
- Intuitive user interface supports menu-driven mission planning and cross-cuing
- Missions developed directly within 3D environment

AUTOMATED SENSOR CONTROL



- Integrated sensor automation and control facilitates rapid mission generation
- Dynamic map footprint provides interactive sensor control

COMMON OPERATIONAL PICTURE (COP) FEEDS AND REAL-TIME SENSOR OVERLAYS

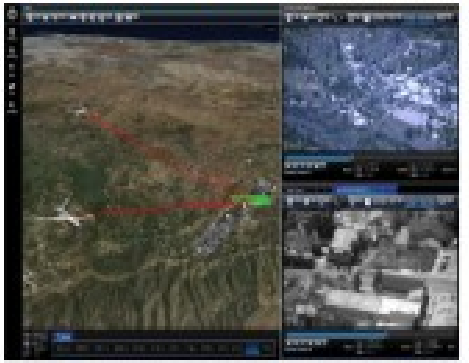


- Enhanced 3D map displays real-time auxiliary feed tracking
- Sensor overlays and aircraft symbology provide improved situational awareness

EXPLOITATION AND DISSEMINATION

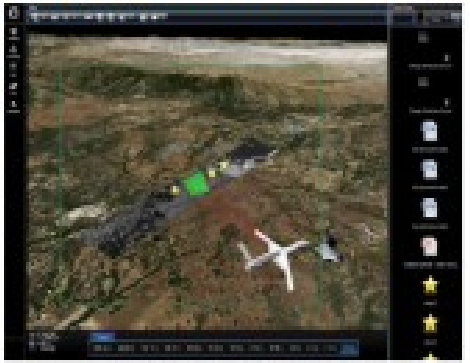
Image Data Manipulation

VIDEO MANAGEMENT SUITE



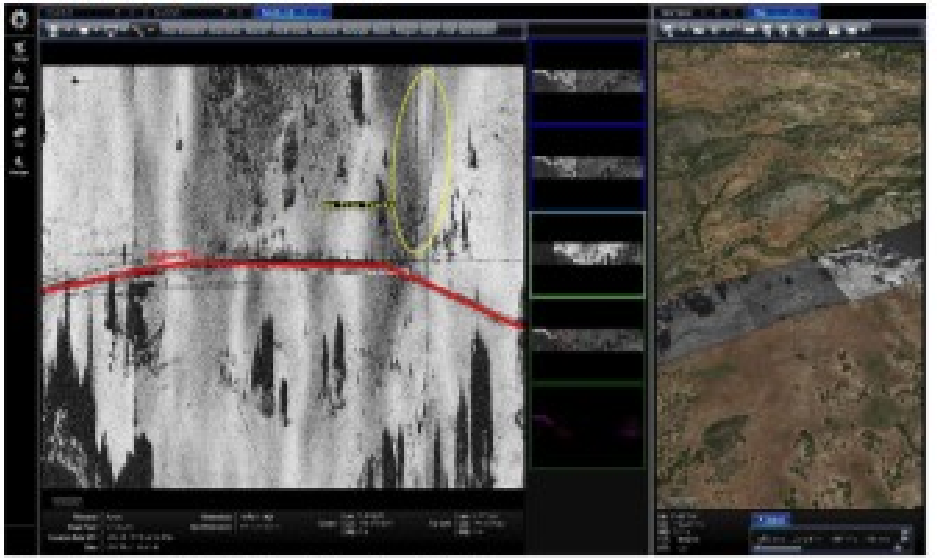
- Simultaneous multiple sensor footprints and feeds increase real-time intelligence product analysis
- Video recording, playback, and annotation tools augment exploitation capabilities

FORENSIC ARCHIVAL AND RETRIEVAL

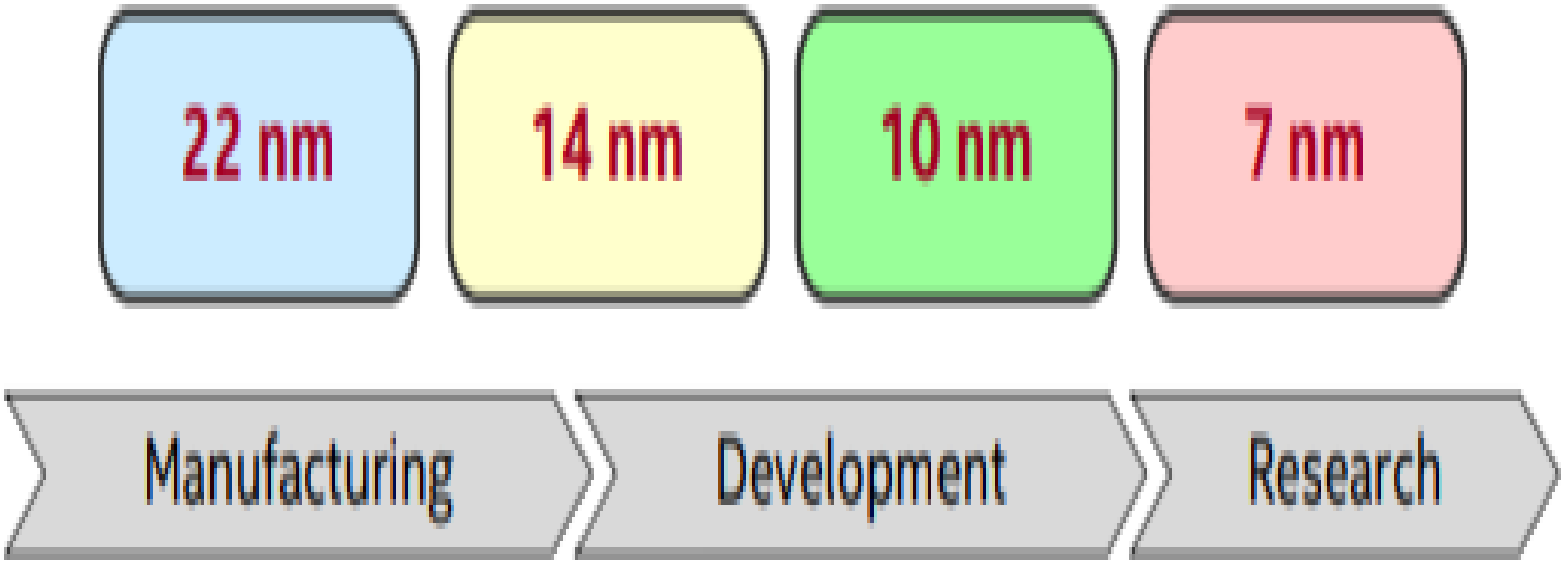


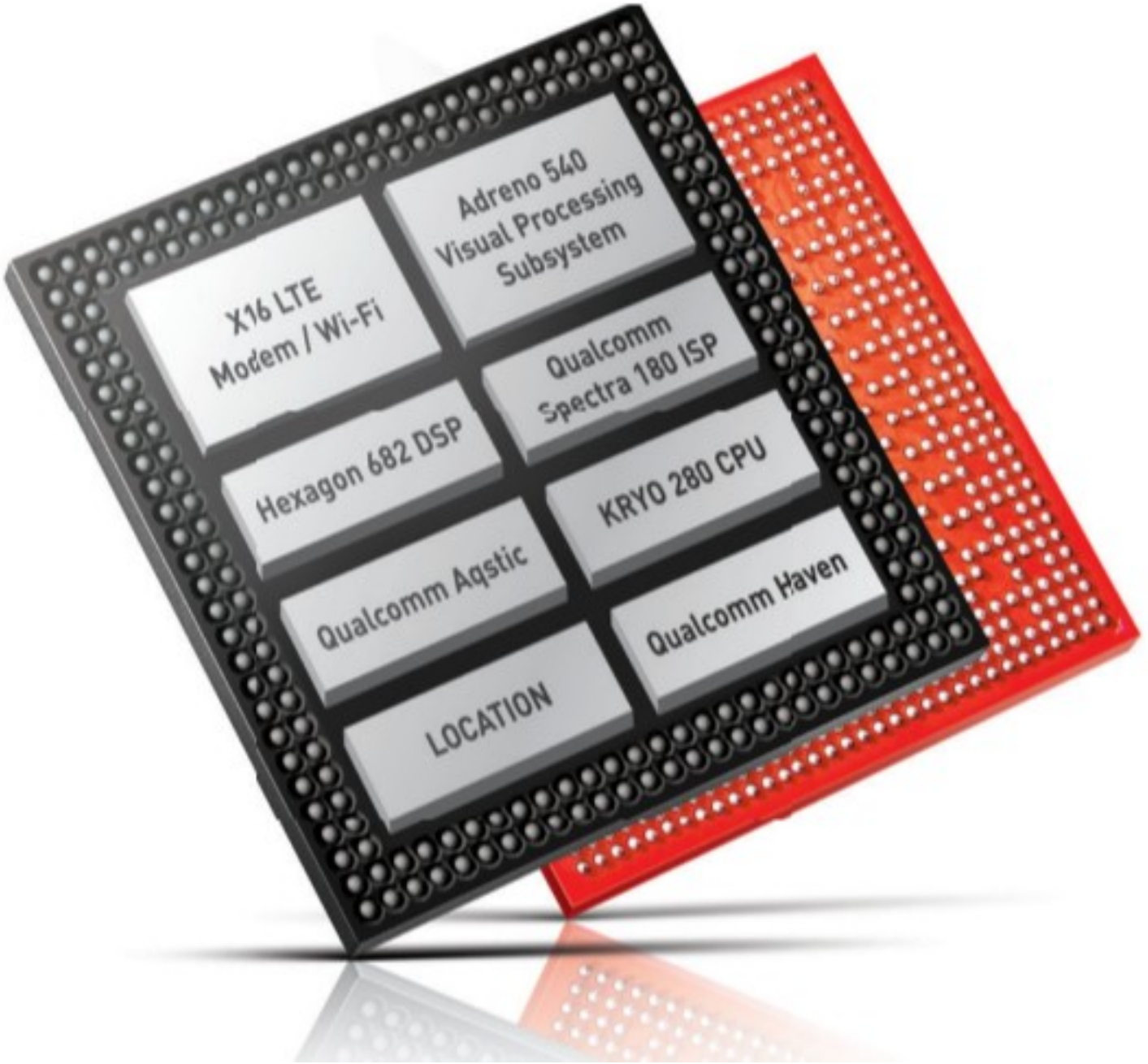
- Integrates System for Tactical Archival, Retrieval, and Exploitation (STARE) server for comprehensive data management
- Perform data queries within federated STARE database
- Access SIGACT and other external intelligence through DCGS network

EXPLOITATION AND DISSEMINATION

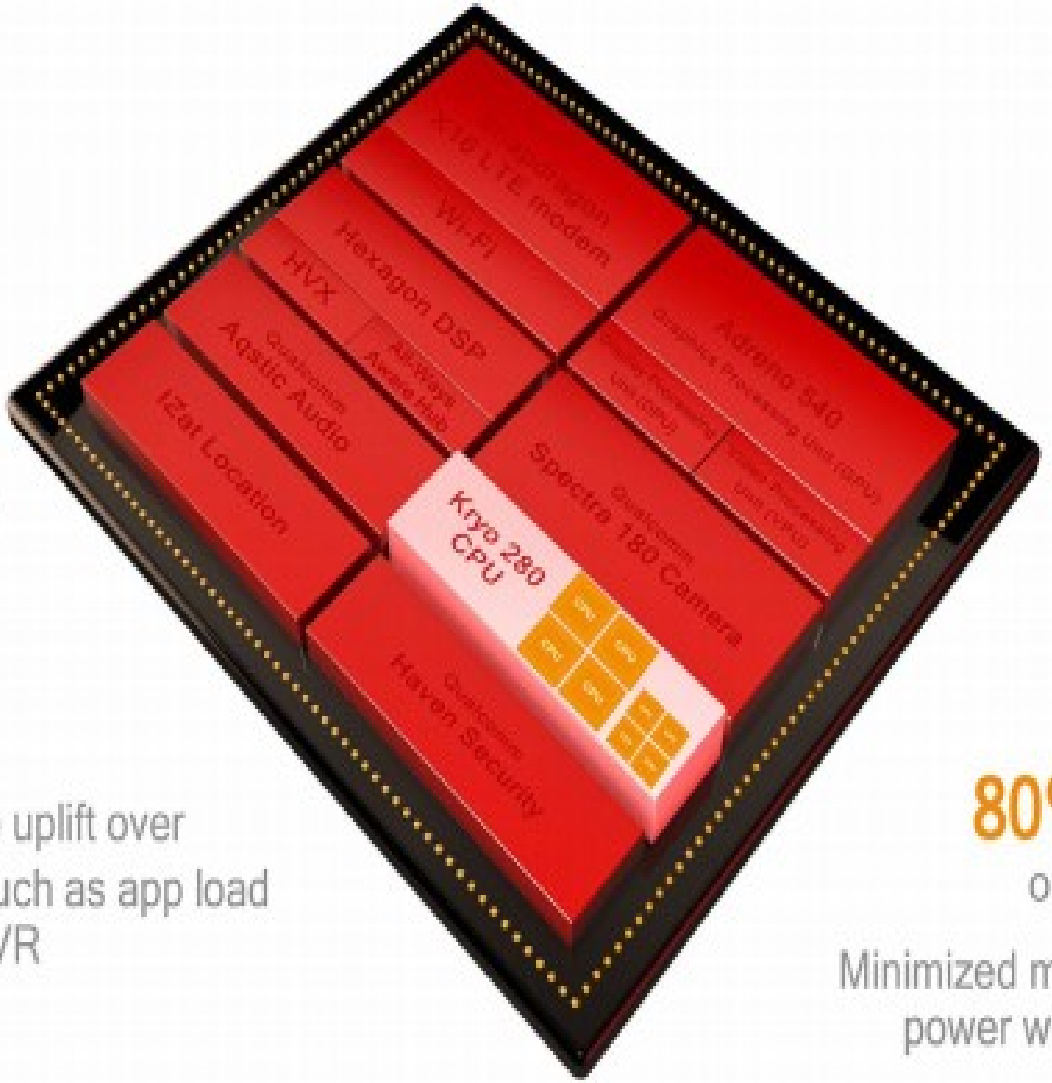


- Create intelligence products with advanced video and exploitation features
- View, cross-reference, and annotate multiple file types for comprehensive analysis
- Disseminate imagery directly from CLAW as GeoPDFs, Powerpoint, GeoTIFF/NITF, and others


22 nm
14 nm
10 nm
7 nm
Manufacturing
Development
Research


X16 LTE
Modem / Wi-Fi
Adreno 540
Visual Processing
Subsystem
Qualcomm
Spectra 180 ISP
Hexagon 682 DSP
KRYO 280 CPU
Qualcomm Aqstic
Qualcomm Haven
LOCATION

Kryo 280 Efficiency Cluster Optimization



CPU CPU
CPU CPU

Performance

Up to 2.45GHz

2MB L2

20% performance uplift over range of use cases such as app load time, web browsing, VR

CPU CPU
CPU CPU

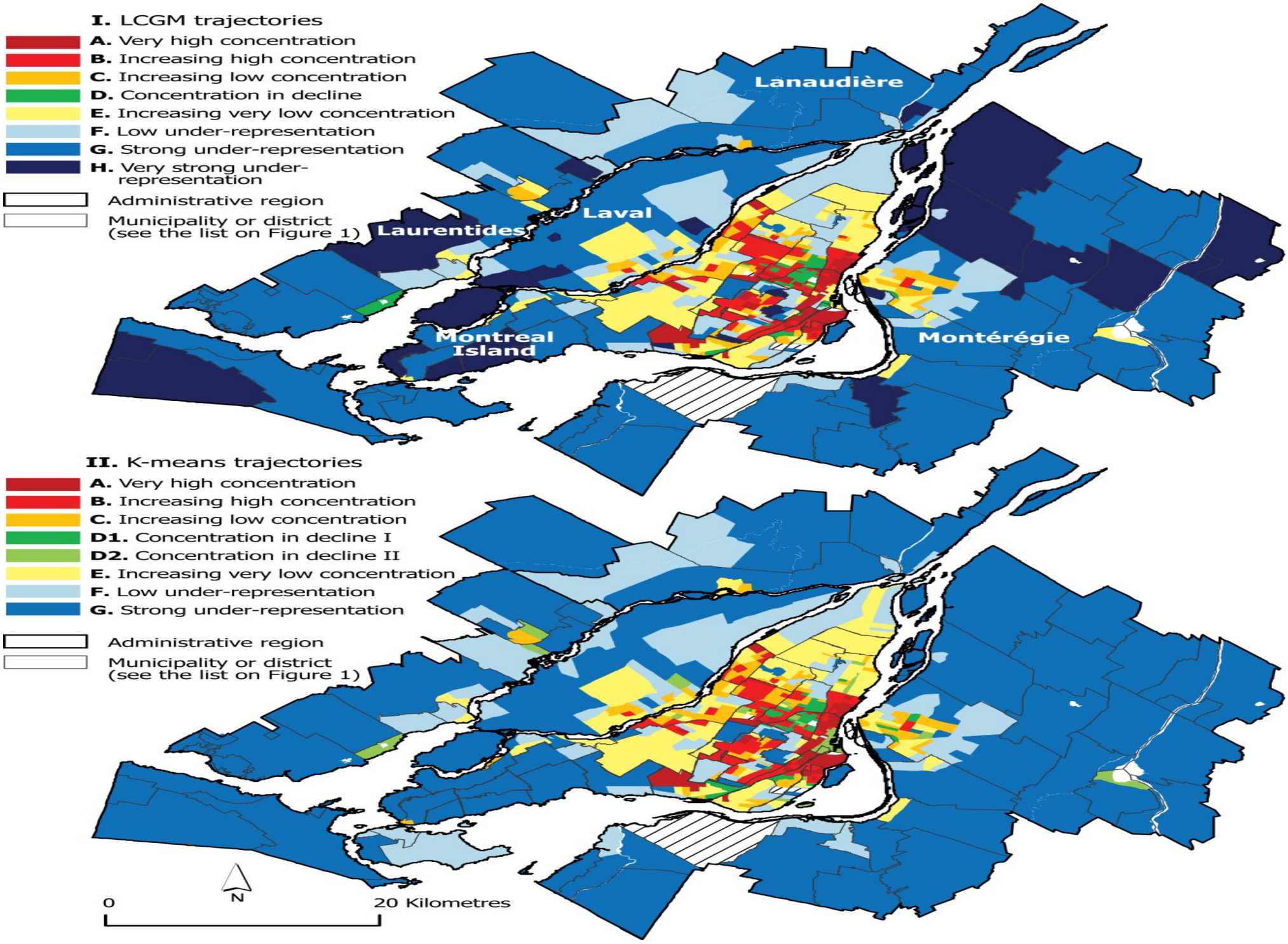
Efficiency

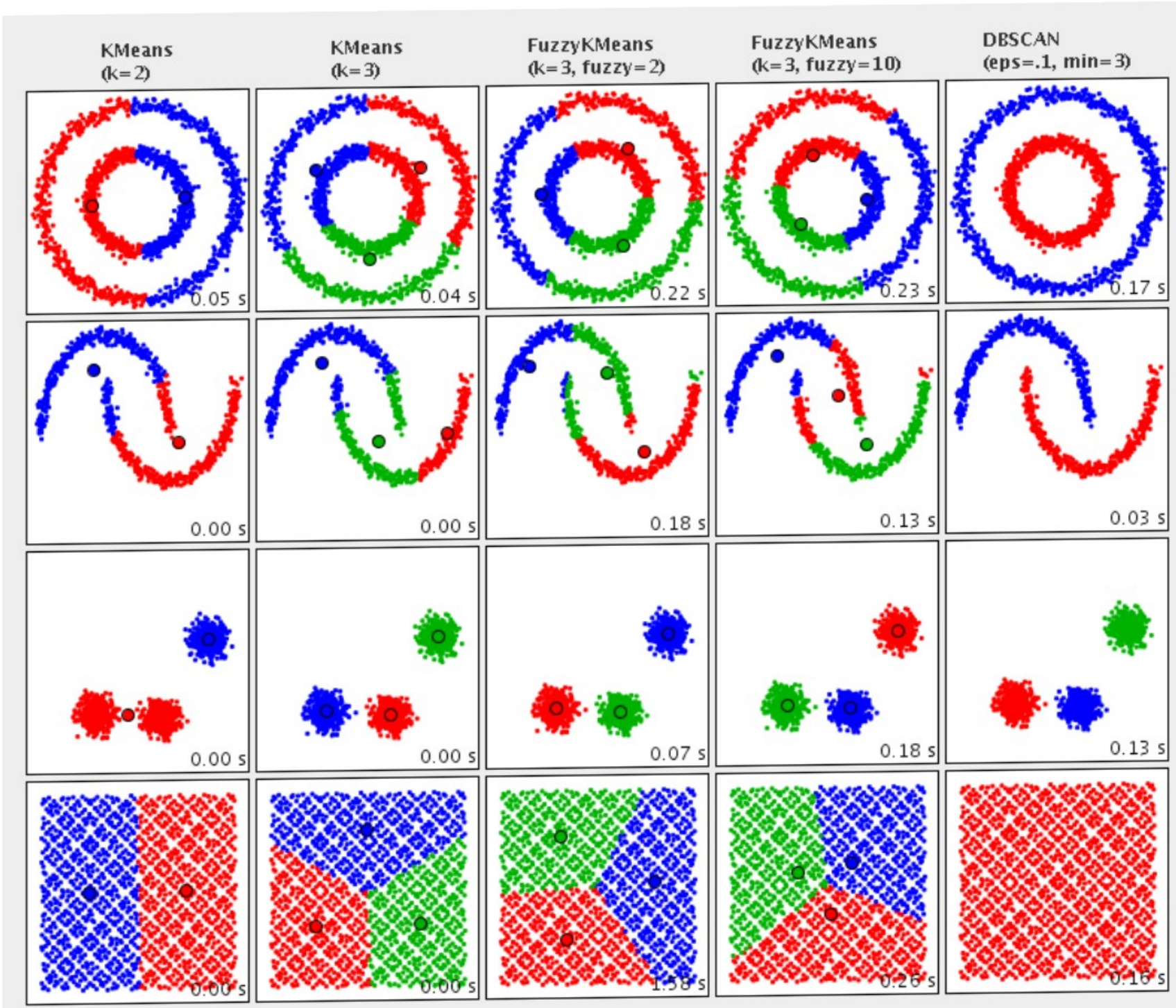
1.9GHz

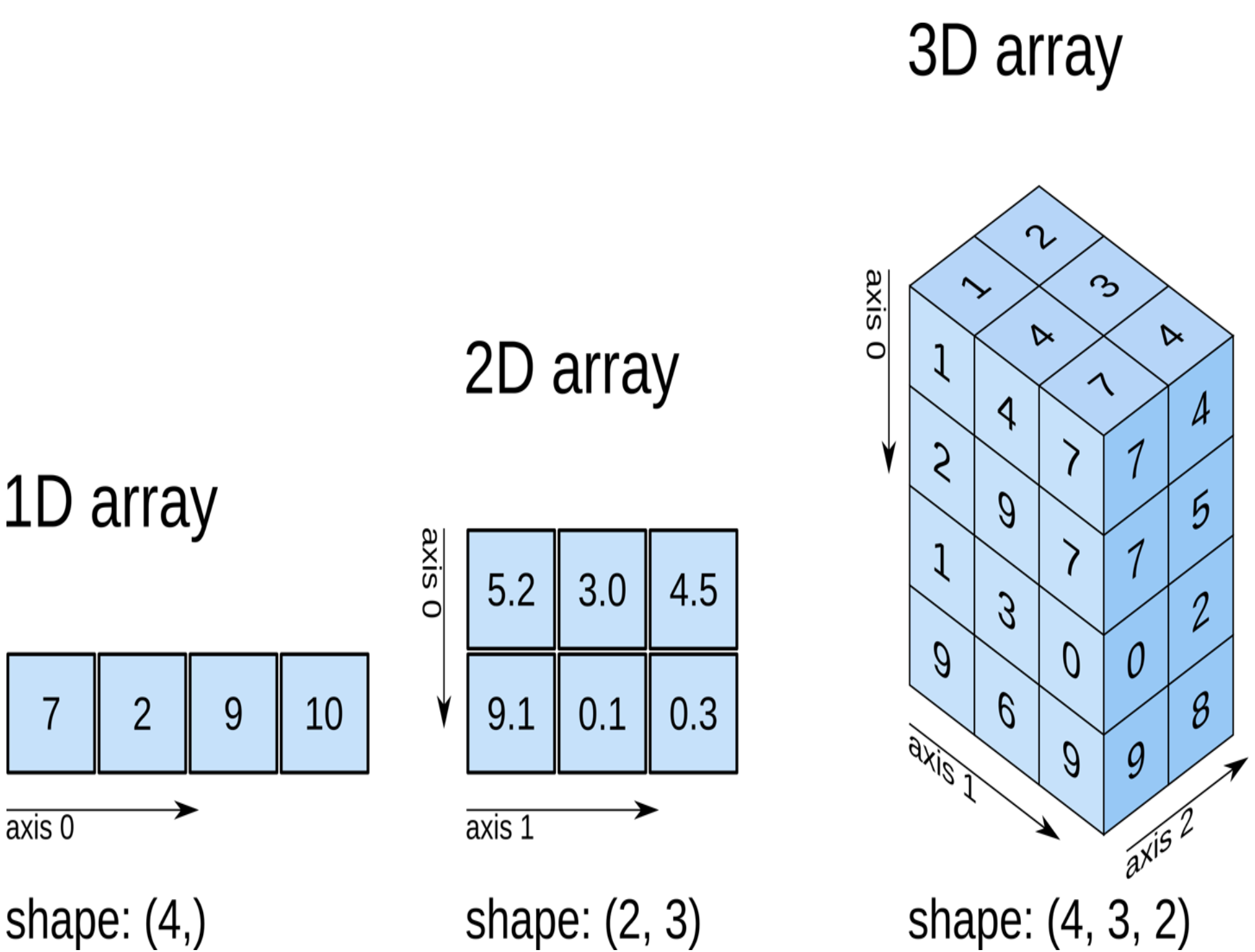
1MB L2

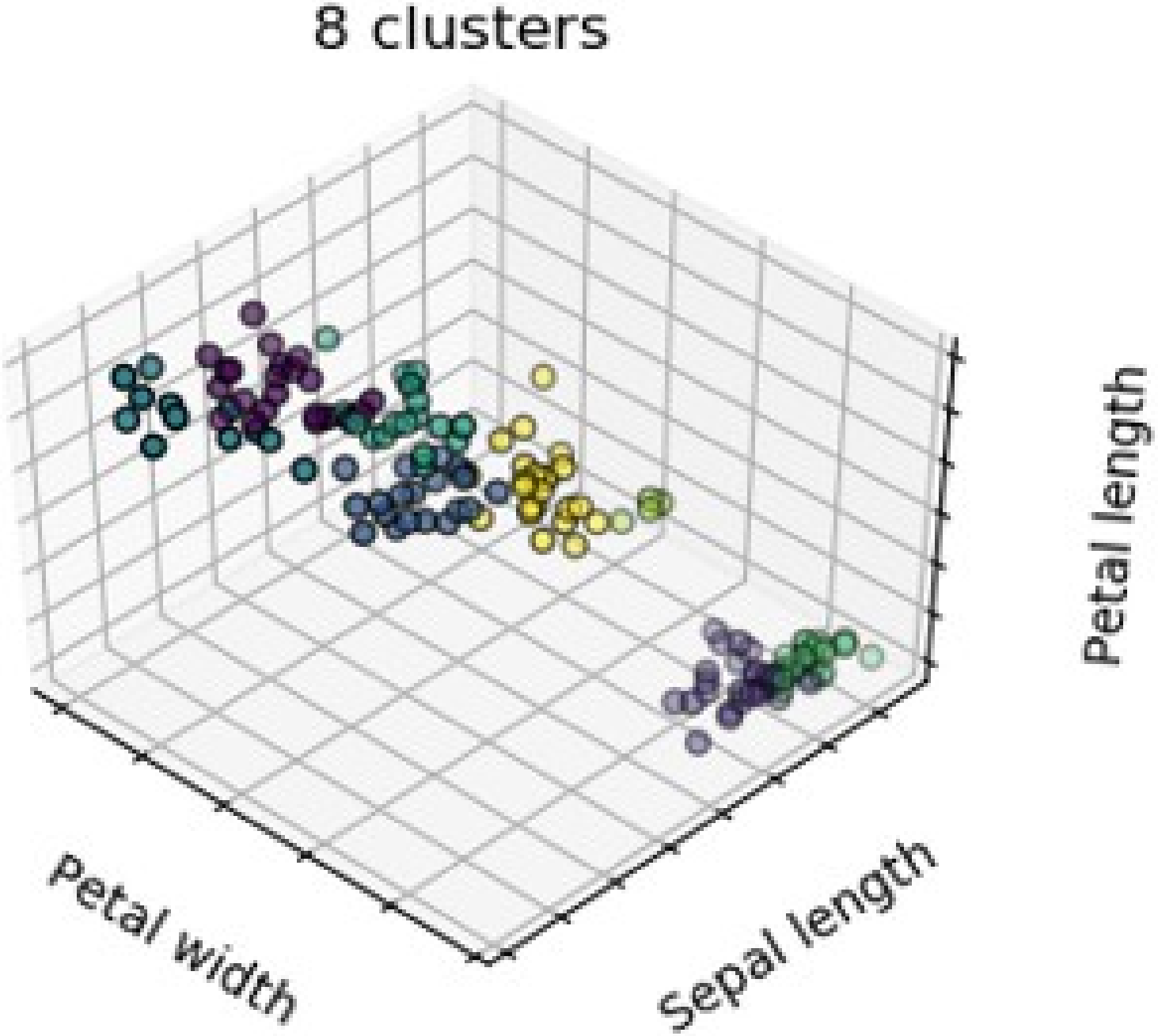
80% of time is spent on efficiency cluster

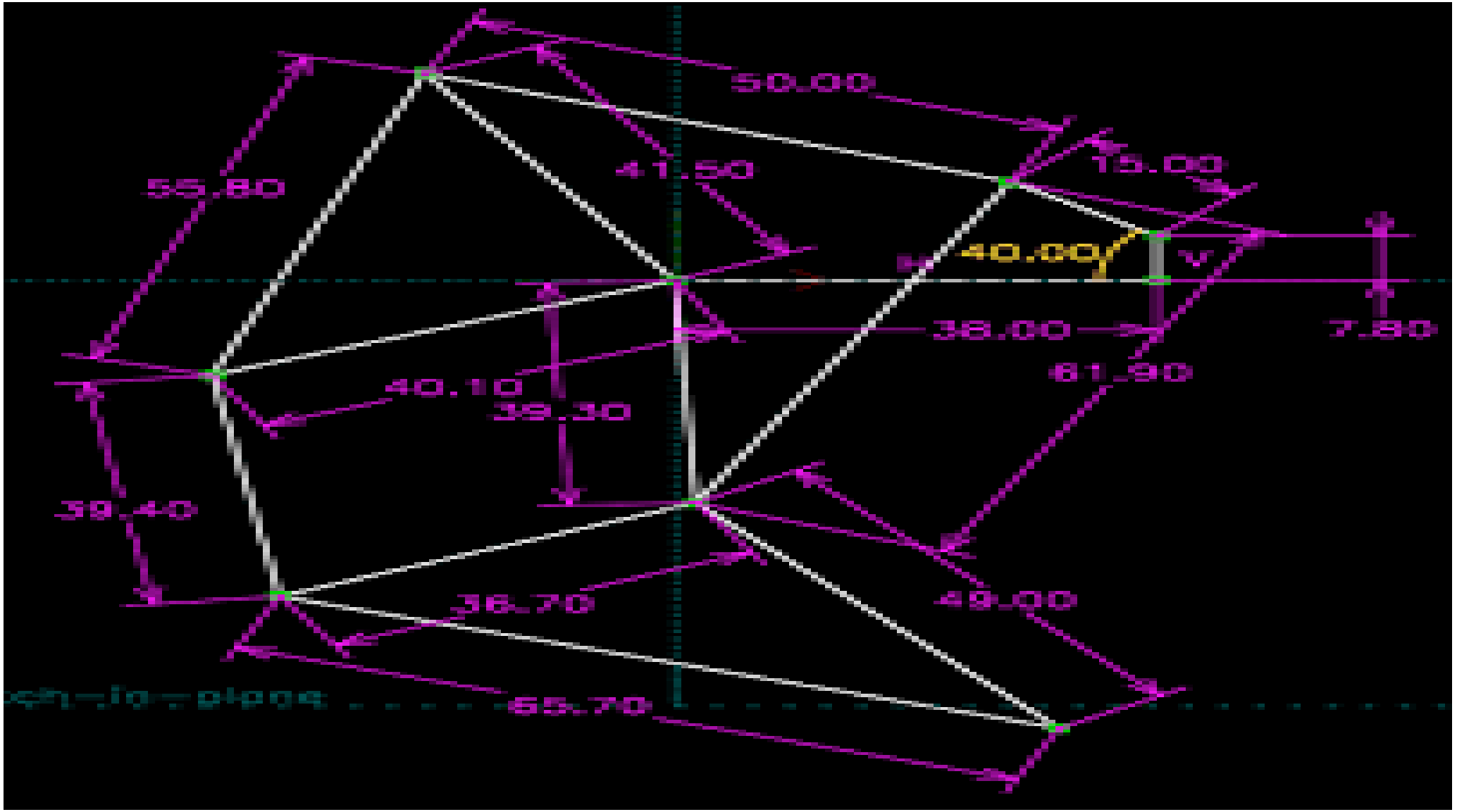
Minimized memory transaction power with larger L2 cache

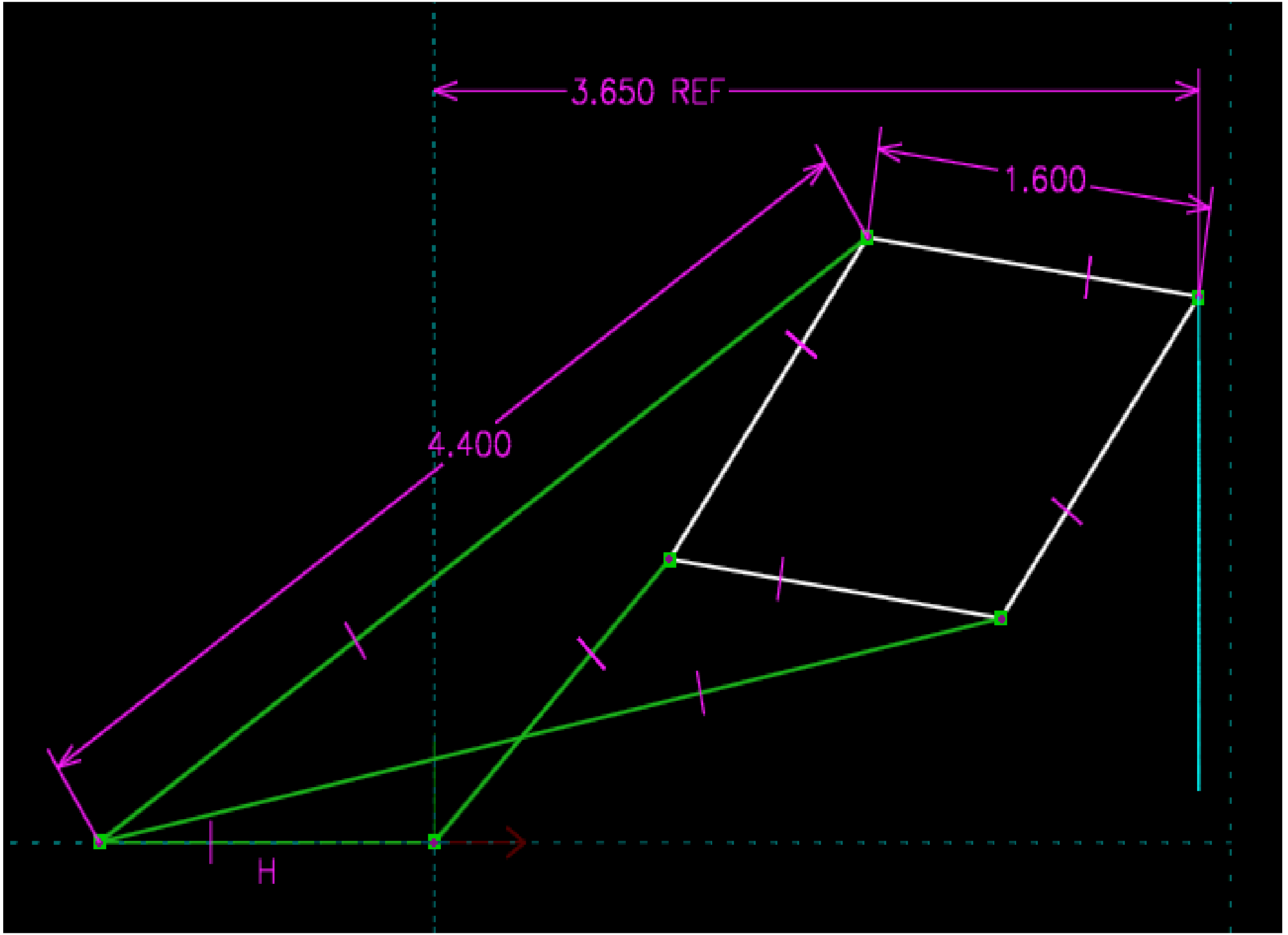

I. LCGM trajectories
A. Very high concentration
B. Increasing high concentration
C. Increasing low concentration
D. Concentration in decline
E. Increasing very low concentration
F. Low under-representation
G. Strong under-representation
H. Very strong under-representation
Administrative region
Municipality or district (see the list on Figure 1)
Lanaudière
Laval
Laurentides
Montreal Island
Montérégie
II. K-means trajectories
A. Very high concentration
B. Increasing high concentration
C. Increasing low concentration
D1. Concentration in decline I
D2. Concentration in decline II
E. Increasing very low concentration
F. Low under-representation
G. Strong under-representation
Administrative region
Municipality or district (see the list on Figure 1)
N
0
20 Kilometres


KMeans
(k=2)
KMeans
(k=3)
FuzzyKMeans
(k=3, fuzzy=2)
FuzzyKMeans
(k=3, fuzzy=10)
DBSCAN
(eps=.1, min=3)
0.05 s
0.04 s
0.22 s
0.23 s
0.17 s
0.00 s
0.00 s
0.18 s
0.13 s
0.03 s
0.00 s
0.00 s
0.07 s
0.18 s
0.13 s
0.00 s
0.00 s
1.58 s
0.26 s
0.16 s


1D array
7
2
9
10
axis 0
shape: (4,)
2D array
axis 0
5.2
3.0
4.5
9.1
0.1
0.3
axis 1
shape: (2, 3)
3D array
axis 0
axis 1
axis 2
shape: (4, 3, 2)


8 clusters
Petal width
Sepal length
Petal length


50.00
55.80
41.50
15.00
40.00
7.80
38.00
61.90
40.10
39.30
39.40
36.70
49.00
65.70


3.650 REF
1.600
4.400
H

```
#Step 4: Taking mean and repeat
c_old=deepcopy(centroid)
for i in range(len(belongs_to)):
    if belongs_to[i]==0:
        mean[0][0]=np.mean(x[i][0])
        mean[0][1]=np.mean(x[i][1])
    else:
        continue

print("New Centroid for cluster 1:",mean[0])
```

```python
import numpy as np

def cluster_points(X, mu):
    clusters  = {}
    for x in X:
        bestmukey = min([(i[0], np.linalg.norm(x-mu[i[0]])) \
                    for i in enumerate(mu)], key=lambda t:t[1])[0]
        try:
            clusters[bestmukey].append(x)
        except KeyError:
            clusters[bestmukey] = [x]
    return clusters

def reevaluate_centers(mu, clusters):
    newmu = []
    keys = sorted(clusters.keys())
    for k in keys:
        newmu.append(np.mean(clusters[k], axis = 0))
    return newmu

def has_converged(mu, oldmu):
    return (set([tuple(a) for a in mu]) == set([tuple(a) for a in oldmu])

def find_centers(X, K):
    # Initialize to K random centers
    oldmu = random.sample(X, K)
    mu = random.sample(X, K)
    while not has_converged(mu, oldmu):
        oldmu = mu
        # Assign all points in X to clusters
        clusters = cluster_points(X, mu)
        # Reevaluate centers
        mu = reevaluate_centers(oldmu, clusters)
    return(mu, clusters)
```