대학생 캠프 강의자료

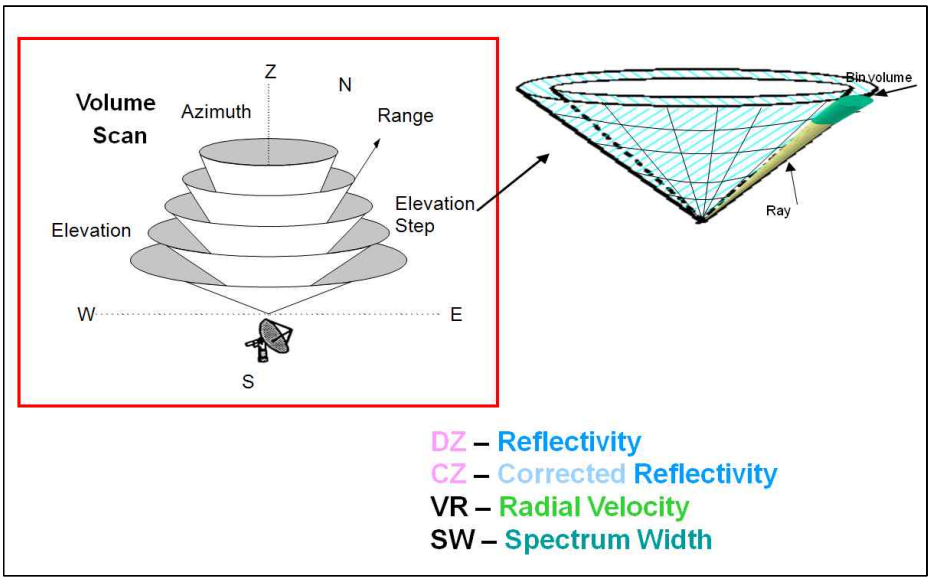
기상레이더센터

KMA Weather Radar Center

레이더자료 처리 실무

제1장 레이더 자료일반

레이더 자료는 스캔한 자료를 순차적으로 저장된 아래 그림1-1과 같이 각 고도(Elevation), ray(Azimuth), bin(sample거리)으로 구성된 3차원자료이다



[그림 1-1 레이더 스캔]

레이더 원시 Volume자료는 제작사마다 형식이 다르지만 기상청에서 사용하는 레이더자료는 UF(Universal Format)자료를 사용하며 2008년 5월부터 저장되어 있다.

UF자료는 Header(Mandatory, Optional, Data, Field), data(DZ, VR, SW, CZ, ...)에 정보를 담고 있는 연속적인 자료이다

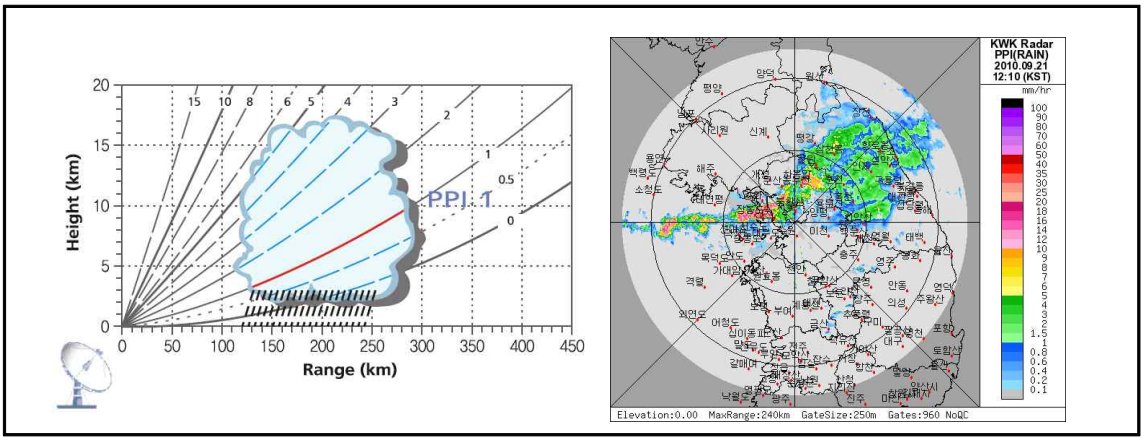
UF자료처리는 RSL(Radar Software Library) Library를 사용하면 쉽게 처리 할수 있으며 이 소프트웨어는 웹사이트에 공개되어 있다.

제2장 레이더 자료처리

2.1 레이더 자료 산출물

2.1.1 PPI(Plan Position Indicator)

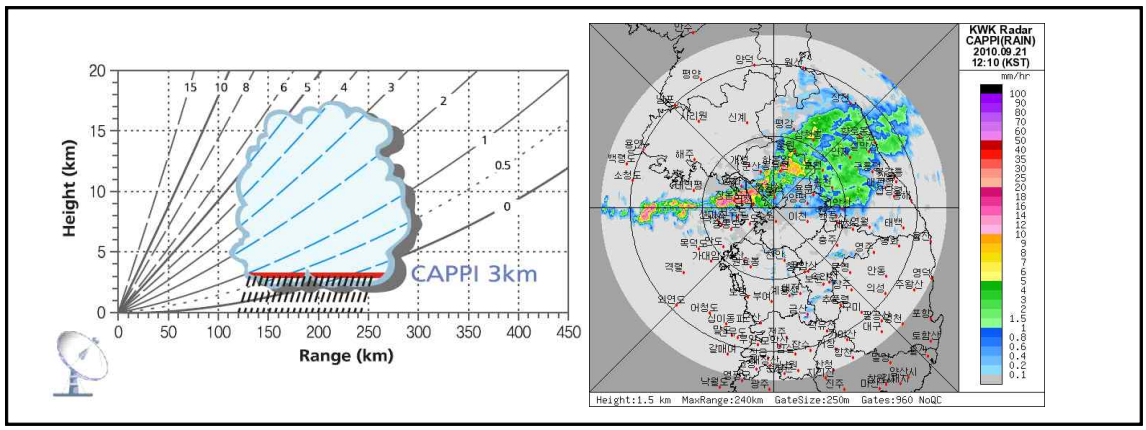
PPI는 주어진 고도각에서 관측된 레이더 에코를 평면 상에 표출한 것이다. PPI는 레이더 관측거리가 멀어질수록 관측고도각이 높아진다.



[그림2-1 PPI 영상]

2.1.2 CAPPI(Constant Altitude PPI)

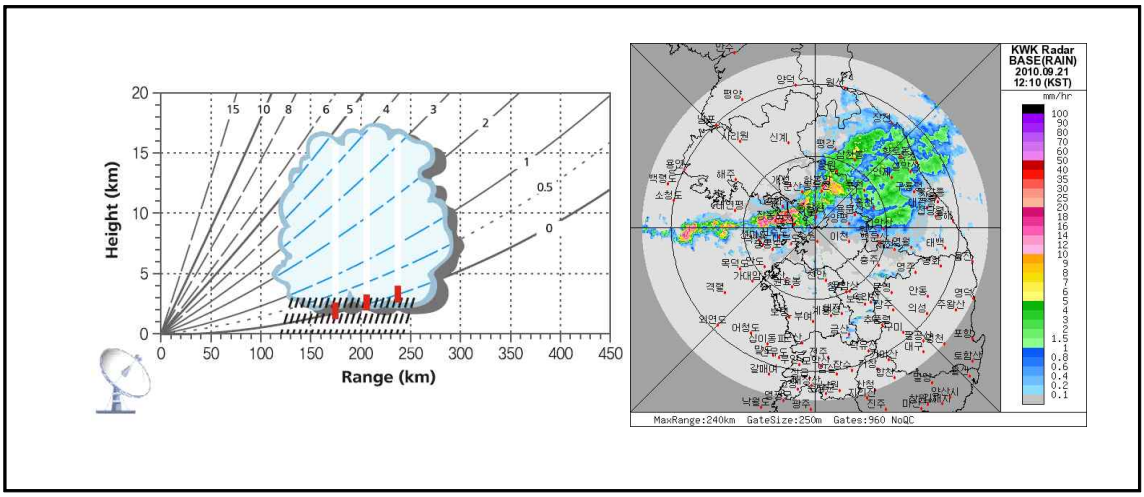
CAPPI는 여러층의 PPI 관측자료를 저장하여 관심 있는 일정한 고도의 수평자료를 뽑아서 표출한 것이다



[그림2-2 CAPPI 영상]

2.1.3 Base Section

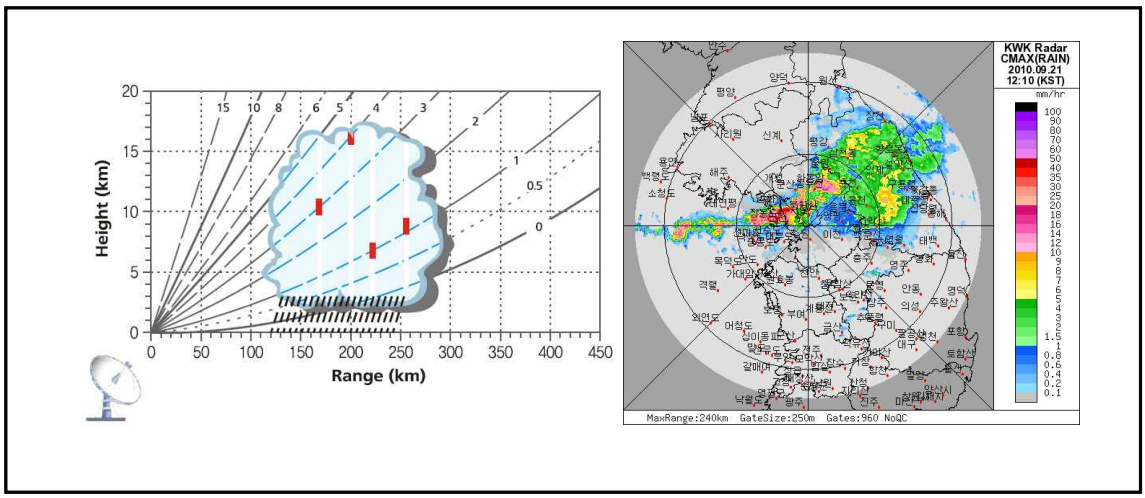
Base Section은 주어진 고도각에 대해서 가장 낮은 높이에서 관측된 에코를 평면상에 나타낸 것이다. 가장 낮은 높이의 자료이므로 지상강수로 환산하거나 비교에 사용된다.



[그림2-3 BASE 영상]

2.1.4 CMAX(Column Max)

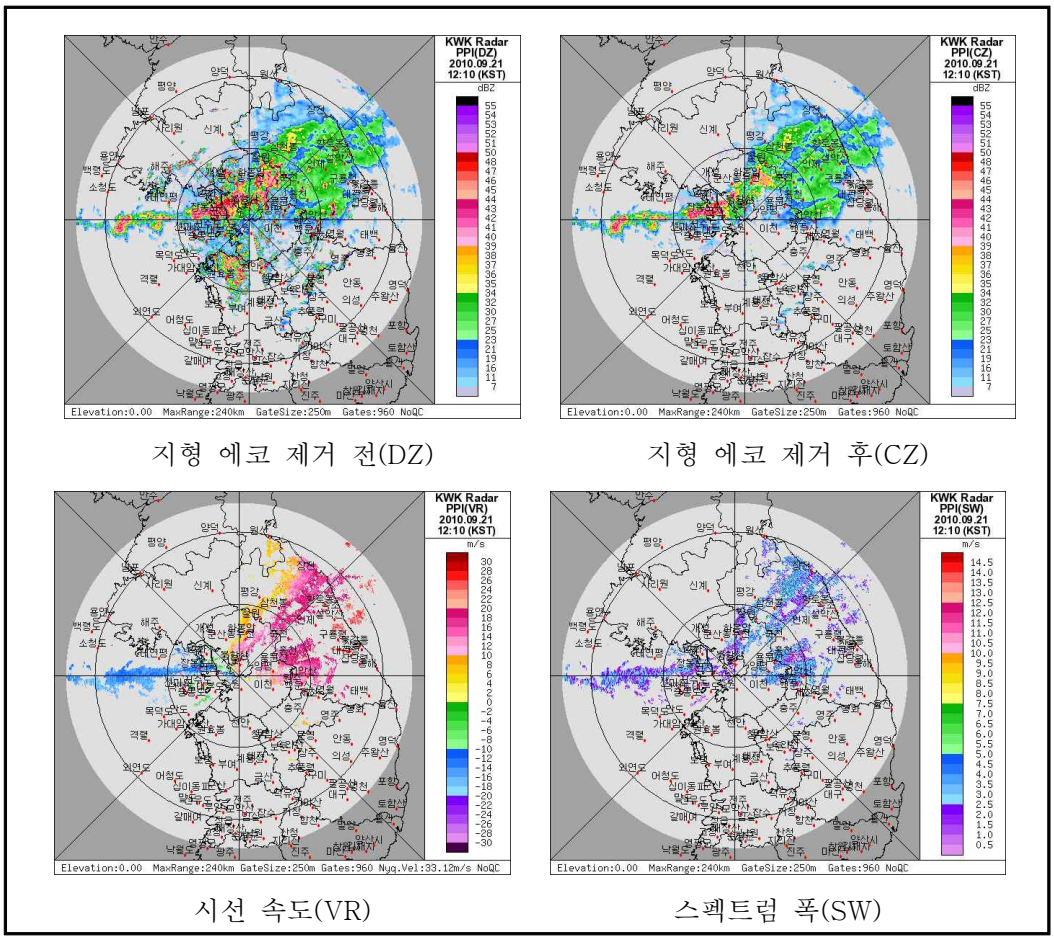
CMAX는 관측된 에코가운데서 연직으로 가장 강한 에코를 평면상에 표출한 것이다.



[그림2-4 CMAX 영상]

2.1.5 사이트 영상

사이트영상은 레이더 사이트별로 지형에코 제거 전(DZ) 후(CZ)의 반사도, 시선속도(VR), 스텍트럼 폭(SW) 영상을 제공하여 에코(구름)의 강도 이동 방향 등을 분석한다.



지형 에코 제거 전(DZ)　　지형 에코 제거 후(CZ)

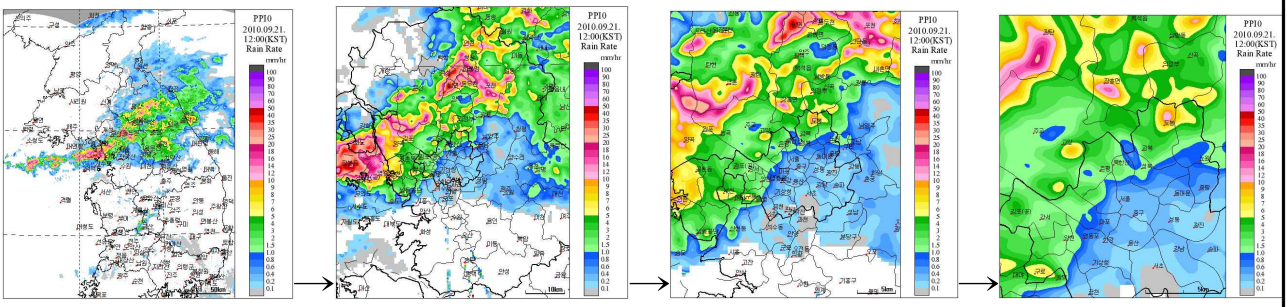
시선 속도(VR)　　스펙트럼 폭(SW)

[그림2-5 사이트 영상]

2.1.6 합성 영상

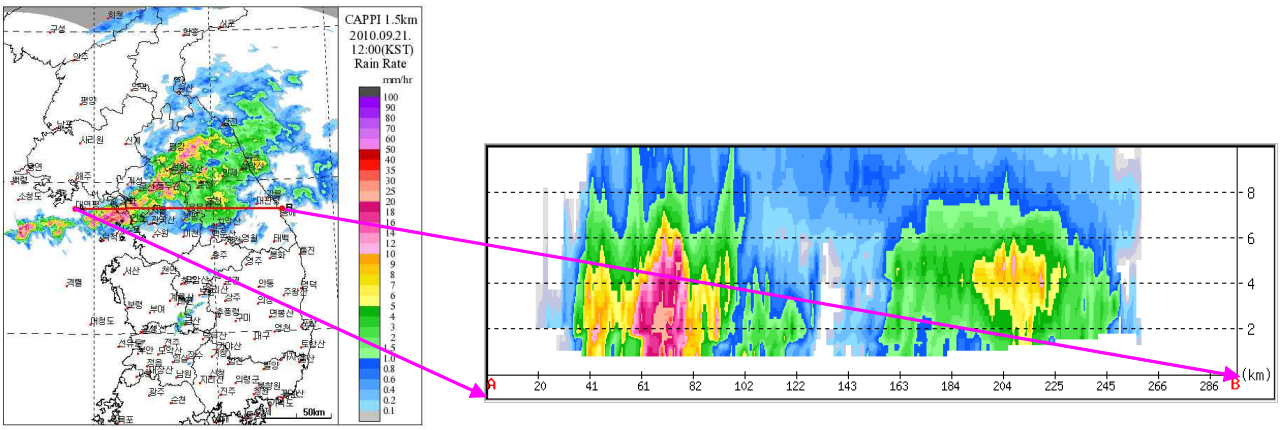
합성영상은 레이더 사이트의 자료들을 합성하여 표출 제공하여 에코(구름)를 확대, 축소, 연직단면 등 여러 형태로 분석한다.

합성영상 확대축소

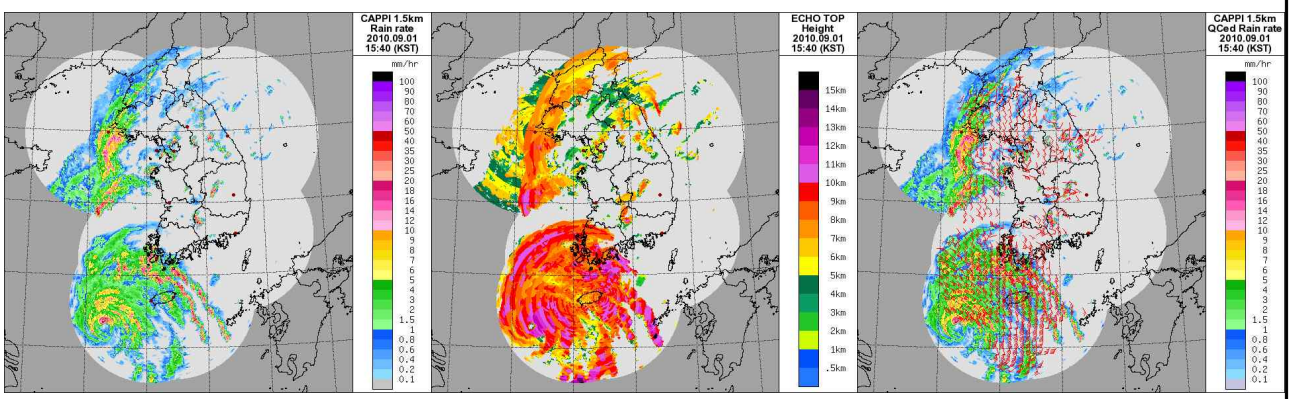


확대 전 확대 1 확대 2 확대 3

연직 단면



합성 영상 표출 분석



강수분포 에코 높이 이중바람장

[그림2-5 합성 영상]

2.2 레이더 자료처리

o 레이더 자료는 원시 Volume자료(UF : Universal Format)를 바탕으로 합성을 하여 이진 및 영상으로 변환한다.

o 이 원시 Volume자료를 처리하려면 RSL(Radar Software Library)를 숙지 해야한다. 관련자료는 아래 Web싸이트에 있다.

http://trmm-fc.gsfc.nasa.gov/trmm_gv/software/rsl/index.html

o 다음으로는 지도 Mapping으로 합성자료는 Lambert conformal conic, 싸이트자료는 Azimuthal equidistant 방법을 사용한다

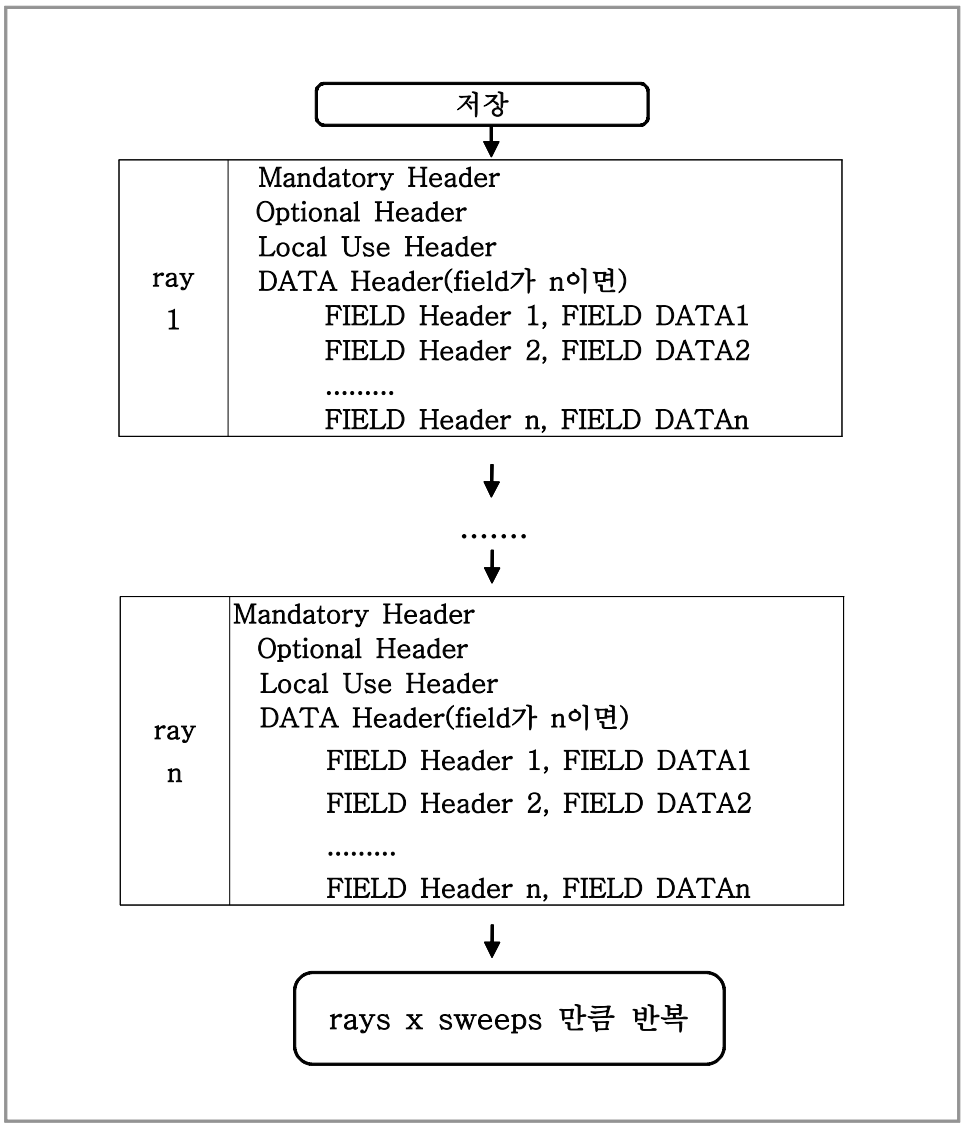
o 이와 관련 자료처리를 하려면 기본적으로 아래 Library를 설치해야 한다.

1. RSL : ftp://trmm-fc.gsfc.nasa.gov/software
2. gd : http://www.libgd.org
3. zlib : http://www.zlib.net
4. png : http://www.libpng.org/pub/png/libpng.html
5. jpeg : http://quox.org/install/graphics/jpeg-6b.html
6. freetype : http://freetype.sourceforge.net/index2.html

2.3 UF자료 구조

UF자료는 레이더에서 scan한 ray순으로 Ray의 갯수 x Sweep의 개수 만큼 저장되어 있다

ray는 Mandatory, Optional, Local Use, DATA, FIELD Header와 FIELD DATA로 구성되고 Optical Header는 경우에 따라 사용되며 Local Use Header는 거의 사용 안된다.



[그림2-7 UF자료 저장]

2.3.1 UF자료(비슬산 이중편파레이더)

```
radar@radar3:/RDPS/KWG/ufview$ ./ufview RDR_BSL_201111220012.uf|more
iCount = 4
Master record length:14926/1
 1:001: Az:359.77, El:-0.50, Time:15:12:54
==========UF Mandatory Header(90)=========
                    ID = UF
           Record size = 7463 words
 Optional Header Pos = 46
Local Use Header Pos = 60
     Data Header Pos = 60
        RecordNumber = 1
        VolumeNumber = 1
           RayNumber = 1
       Record in Ray = 1
        Sweep Number = 1
           RadarName = BISLSAN_
            SiteName = 'BISLSAN_#'
                 Lat = 035 41 39.47
                 Lon = 128 32 6.72
            Altitude = 1085 m
         Date & Time = 2011-11-21 15:12:54 UTC
           Time Zone = UT
             Azimuth = -0.23
           Elevation = -0.50
          Sweep Mode = 1
         Fixed angle = -0.50
          Sweep Rate = 768
        Convert Date = 2011-11-21
        Convert Name = IRIS8.11
       No Data Value = 0x8000
==========UF Optional Header(28)=========
         ProjectName = TEST
     BaselineAzimuth = NO-DATA
   BaselineElevation = NO-DATA
     VolumeScan Time = 15:12:33
       FieldTapeName = Default
                Flag =  1
==========UF Data Header(70)=========
  Fields in this ray = 9
 Records in this ray = 1
  Fields this record = 9
      Field 000 type = DZ
           Position  = 81
      Field 001 type = CZ
           Position  = 901
      Field 002 type = VR
           Position  = 1721
      Field 003 type = SW
           Position  = 2543
      Field 004 type = DR
           Position  = 3363
      Field 005 type = KD
           Position  = 4183
      Field 006 type = PH
           Position  = 5003
      Field 007 type = SQ
           Position  = 5823
      Field 008 type = RH
           Position  = 6643
==========UF Field Header(66) for DZ=========
      Data position  = 100 words
       Scale factor  = 100
```

```
      Starting range = 0.000 km
         Bin spacing = 250 meters
           Bin count = 801
         Pulse width = 600 meters
  Horiz. beam width = 0.97 deg
Vertical beam width = 0.97 deg
 Receiver bandwidth = 600 MHz
       Polarization = Horizontal
         Wavelength = 10.75 cm
        Sample size = 64
     Threshold data =
    Threshold value = -32768
              Scale = -32768
          Edit code =
                PRT = 2000 microseconds (500.0 Hertz)
       Bits per bin = 16

===UF Field Data(8400) for DZ====
   0:  1.0 -11.0 -12.5  -8.5  18.5  13.0   3.0   8.5  12.5  15.5  21.5   4.0   0.5   5.0   7.5
  15: -4.0   7.0  14.5  20.5  19.5  -5.0  -5.0  -2.5  -1.0   2.0  -5.0  -0.5  -5.5   2.0  -0.5
  30:  2.5  -7.0   0.5  -2.0  -2.0  -4.5  -4.5  -0.5   1.0   5.5  -2.0   4.5   0.5  -1.5 -10.5
  45:  0.0  -9.0 -16.0  -8.5   0.0   0.5   3.0  -3.5  -1.5   3.5  -9.5  -0.5   1.5  16.5  -0.5
  60:-12.0   9.0  -2.0  -5.0  10.5   3.5   1.0  19.0  18.5  24.5  23.0  27.5  22.5  31.5  35.5
  75: 34.5  38.0  43.0  31.5  25.0  29.0  27.5  21.0  25.5  28.0  26.0  25.0  20.0  14.5  22.5
      ~ 중략 ~
```

2.3.2 ufview 소스

```
/* ==========================================================================
 * project : ufview
 * gcc -o ufview ufview.c -L/usr/local/trmm/GVBOX/lib -lm -lrsl -lfl
 * ==========================================================================*/

#include <locale.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <zlib.h>
#include <zconf.h>

#include "./sigtypes.h"
#include "./uf_header.h"

static char *sTailString(const char *sString_a, int iMaxSize_a );
static void swap2(void *word);
static void swap4(void *word);
static void swap2_array(short *buf, int iCount_a );

/* ============================================================ */
int main(int argc, char *argv[])
{
  int iCount, iRecordLength, iReadRayCount, iPosition;
  struct uf_mandatory_header2 MaHeader;
  struct uf_optional_header    OpHeader;
  struct uf_data_header2       DataHeader;
  FILE *stream;

  int iSeekSize, iMoment;
  FLT4 fAz, fEl;
  struct uf_field_header2 FieldHeader;

  int iBinCount;
  short iDataArray[4200];

  int iBin, iValue;
  int iTotalLength, iThisLength;
  float  fScale;
  char sThreshold[20];
```

```
char sFormat[20];
char sThisString[20];


stream = gzopen(argv[1], "r" );

if (!stream)  exit( EXIT_FAILURE);

/* Loop reading in mandatory headers, and displaying the results */
iReadRayCount = 0;

iCount = gzread(stream, &iRecordLength, 4);
iPosition = 0;

printf("iCount = %d ₩n",iCount);


while(1) //( iCount == 4 )
{
    if (LITTLE_END) swap4( &iRecordLength );
        iReadRayCount++;

    if ( iCount == 0 ) goto DONE;
    if (iReadRayCount > iRecordLength) goto DONE;

    printf( "Master record length:%d/%d₩n", iRecordLength,iReadRayCount );

    /*************************************************************************
     *        Read and process mandatory header                              *
     *************************************************************************/
    iCount = gzread( stream, &MaHeader, sizeof(MaHeader));
    iPosition += sizeof(MaHeader);

    if (LITTLE_END)
    {
        //char  sID[2];
        swap2( &MaHeader.iRecordSize);
        swap2( &MaHeader.iOptionalHeaderPosition);
        swap2( &MaHeader.iLocalUseHeaderPosition);
        swap2( &MaHeader.iDataHeaderPosition);
        swap2( &MaHeader.iRecordNumber );
        swap2( &MaHeader.iVolumeNumber );
        swap2( &MaHeader.iRayNumber );
        swap2( &MaHeader.iRecordInRay );
        swap2( &MaHeader.iSweepNumber );
        //char  sRadarName[8];
        //char  sSiteName[8];
        swap2( &MaHeader.iLatDegrees );
        swap2( &MaHeader.iLatMinutes );
        swap2( &MaHeader.iLatSeconds );
        swap2( &MaHeader.iLonDegrees );
        swap2( &MaHeader.iLonMinutes );
        swap2( &MaHeader.iLonSeconds );
        swap2( &MaHeader.iAltitude );
        swap2( &MaHeader.iYear );
        swap2( &MaHeader.iMonth );
        swap2( &MaHeader.iDay );
        swap2( &MaHeader.iHour );
        swap2( &MaHeader.iMinute );
        swap2( &MaHeader.iSecond );
        //char  sTimeZone[2];
        swap2( &MaHeader.iAzimuth );
        swap2( &MaHeader.iElevation );
        swap2( &MaHeader.iSweepMode );
        swap2( &MaHeader.iFixedAngle );
        swap2( &MaHeader.iSweepRate );
        swap2( &MaHeader.iConvertYear );
        swap2( &MaHeader.iConvertMonth );
        swap2( &MaHeader.iConvertDay );
        //char  sConvertName[8];
        swap2( &MaHeader.iNoDataValue );
    }
```

```
fAz = ((FLT4)MaHeader.iAzimuth)/64.0;
if (fAz<0.0) fAz+= 360.0;
fEl = ((FLT4)MaHeader.iElevation)/64.0;

printf( "%2d:%03d: Az:%6.2f, El:%5.2f, Time:%02d:%02d:%02d₩n",
MaHeader.iSweepNumber, MaHeader.iRayNumber, fAz, fEl,
MaHeader.iHour, MaHeader.iMinute, MaHeader.iSecond );

printf("==========UF Mandatory Header(%d)========= ₩n",sizeof(MaHeader));
printf("                    ID = %-2.2s ₩n",MaHeader.sID);
printf("           Record size = %d words ₩n",MaHeader.iRecordSize);
printf(" Optional Header Pos = %d ₩n",MaHeader.iOptionalHeaderPosition);
printf("Local Use Header Pos = %d ₩n",MaHeader.iLocalUseHeaderPosition);
printf("      Data Header Pos = %d ₩n",MaHeader.iDataHeaderPosition);
printf("         RecordNumber = %d ₩n",MaHeader.iRecordNumber);
printf("         VolumeNumber = %d ₩n",MaHeader.iVolumeNumber);
printf("            RayNumber = %d ₩n",MaHeader.iRayNumber);
printf("        Record in Ray = %d ₩n",MaHeader.iRecordInRay);
printf("         Sweep Number = %d ₩n",MaHeader.iSweepNumber);
printf("            RadarName = %-8.8s ₩n",MaHeader.sRadarName);
printf("             SiteName = '%s' ₩n",MaHeader.sSiteName);
printf("                  Lat = %03d %02d %.2f ₩n",MaHeader.iLatDegrees,
                MaHeader.iLatMinutes,(float)MaHeader.iLatSeconds/64.0);
printf("                  Lon = %03d %02d %.2f ₩n",MaHeader.iLonDegrees,
                MaHeader.iLonMinutes,(float)MaHeader.iLonSeconds/64.0);
printf("             Altitude = %d m₩n",MaHeader.iAltitude);
printf("          Date & Time = %04d-%02d-%02d ",MaHeader.iYear,
                                MaHeader.iMonth,MaHeader.iDay);
printf("%02d:%02d:%02d UTC₩n",MaHeader.iHour,
                                MaHeader.iMinute, MaHeader.iSecond);
printf("            Time Zone = %-2.2s ₩n",MaHeader.sTimeZone);
printf("              Azimuth = %5.2f ₩n",(float)MaHeader.iAzimuth/64.0);
printf("            Elevation = %5.2f ₩n",(float)MaHeader.iElevation/64.0);
printf("           Sweep Mode = %d ₩n",MaHeader.iSweepMode);
printf("          Fixed angle = %5.2f ₩n",(float)MaHeader.iFixedAngle/64.0);
printf("           Sweep Rate = %d ₩n", MaHeader.iSweepRate );
printf("         Convert Date = %04d-%02d-%02d ₩n",MaHeader.iConvertYear,
                        MaHeader.iConvertMonth,MaHeader.iConvertDay);
printf("         Convert Name = %-8.8s ₩n",MaHeader.sConvertName);
printf("        No Data Value = 0x%04hX₩n",MaHeader.iNoDataValue);

/*****************************************************************
 *        Read and process optional header                       *
 *****************************************************************/
if (MaHeader.iOptionalHeaderPosition != MaHeader.iLocalUseHeaderPosition)
{
      iCount = gzread( stream, &OpHeader, sizeof(OpHeader) );
      iPosition += sizeof(OpHeader);

      if (LITTLE_END)
      {
            //char   sProjectName[8];
            swap2(   &OpHeader.iBaselineAzimuth);
            swap2(   &OpHeader.iBaselineElevation);
            swap2(   &OpHeader.iVolumeScanHour);
            swap2(   &OpHeader.iVolumeScanMinute);
            swap2(   &OpHeader.iVolumeScanSecond);
            //char   sFieldTapeName[8];
            swap2(   &OpHeader.iFlag);
      }

       printf("==========UF Optional Header(%d)=========₩n",sizeof(OpHeader));
       printf("           ProjectName = %-8.8s ₩n",OpHeader.sProjectName);

       if (OpHeader.iBaselineAzimuth != MaHeader.iNoDataValue)
            printf("       BaselineAzimuth = %5.2f ₩n",(float)OpHeader.iBaselineAzimuth/64.0); // z < 0 , z +=360.0
       else
            printf("       BaselineAzimuth = NO-DATA ₩n");

       if (OpHeader.iBaselineElevation != MaHeader.iNoDataValue)
            printf("     BaselineElevation = %5.2f ₩n",(float)OpHeader.iBaselineElevation/64.0);
```

```
        else
            printf("    BaselineElevation = NO-DATA ₩n");

        printf("       VolumeScan Time = %02d:%02d:%02d ₩n",OpHeader.iVolumeScanHour,
                                OpHeader.iVolumeScanMinute,OpHeader.iVolumeScanSecond);
        printf("          FieldTapeName = %-8.8s ₩n",OpHeader.sFieldTapeName);
        printf("                   Flag =  %d ₩n",OpHeader.iFlag);

}

/*********************************************************************
*         ReadDataHeader                                             *
 *********************************************************************/
/* Read and process data header, maybe skip local use header first */
iSeekSize = 2*(MaHeader.iDataHeaderPosition-1) - iPosition;

if (iSeekSize !=0) iCount = gzseek( stream, iSeekSize, SEEK_CUR );
iPosition += iSeekSize;


iCount = gzread( stream, &DataHeader, 6 );
if (iCount != 6) perror( "fread" );
iPosition += 6;

if (LITTLE_END)
{
    swap2(   &DataHeader.iFieldsThisRay );
    swap2(   &DataHeader.iRecordsThisRay );
    swap2(    &DataHeader.iFieldsThisRecord );
    //struct uf_dsi2 Dsi[UF_MAX_MOMENTS];
}

printf("==========UF Data Header(%d)=========₩n",sizeof(DataHeader));
printf("  Fields in this ray = %d ₩n",DataHeader.iFieldsThisRay );
printf(" Records in this ray = %d ₩n",DataHeader.iRecordsThisRay );
printf("  Fields this record = %d ₩n",DataHeader.iFieldsThisRecord );

if (DataHeader.iFieldsThisRecord > 16)
{
    printf( "Warning: FieldsThisRecord value of %d exceeds max of %d₩n",
            DataHeader.iFieldsThisRecord, 16);
    DataHeader.iFieldsThisRecord = 16 ;
}

/* Now loop through all the moments */
for (iMoment=0; iMoment<DataHeader.iFieldsThisRecord; iMoment++)
{
       iCount = gzread( stream, &DataHeader.Dsi[iMoment], 4 );
       if (iCount != 4) perror( "fread" );
       iPosition += 4;

       if (LITTLE_END) swap2( &DataHeader.Dsi[iMoment].iFieldHeaderPosition );
       printf( "       Field %03d type = %-2.2s ₩n", iMoment, DataHeader.Dsi[iMoment].sDataType );
       printf( "             Position  = %d ₩n", DataHeader.Dsi[iMoment].iFieldHeaderPosition );
}

/*********************************************************************
 *         ReadFieldHeader                                           *
 *********************************************************************/
/* Read and process data headers, maybe skip some local data each time */
for (iMoment=0; iMoment<DataHeader.iFieldsThisRecord; iMoment++)
{
     //struct uf_field_header2 FieldHeader;
     iSeekSize = 2*(DataHeader.Dsi[iMoment].iFieldHeaderPosition-1) - iPosition;
     if (iSeekSize !=0) iCount = gzseek( stream, iSeekSize, SEEK_CUR );
     iPosition += iSeekSize;

     int iReadSize;
     iReadSize = sizeof(&FieldHeader);

     iCount = gzread(stream, &FieldHeader, 38 );
     if (iCount != 38) perror( "fread" );
```

```
iPosition += 38;

if (LITTLE_END)
{
    swap2( &FieldHeader.iDataPosition );        /* Origin 1 offset of data from start of record */
    swap2( &FieldHeader.iScaleFactor );         /* Met units = file value/ScaleFactor */
    swap2( &FieldHeader.iStartRangeKm );
    swap2( &FieldHeader.iStartRangeMeters );
    swap2( &FieldHeader.iBinSpacing );          /* Bin spacing in meters */
    swap2( &FieldHeader.iBinCount );
    swap2( &FieldHeader.iPulseWidth );          /* Pulse width in meters */
    swap2( &FieldHeader.iBeamWidthH );          /* Horizontal Beam width in 1/64 of degree */
    swap2( &FieldHeader.iBeamWidthV );
    swap2( &FieldHeader.iBandWidth );           /* Receiver bandwidth in 1/64 Mhz */
    swap2( &FieldHeader.iPolarization );
    swap2( &FieldHeader.iWaveLength );          /* Wavelength in 1/64 of a cm */
    swap2( &FieldHeader.iSampleSize );          /* Sample size */
    //char  sThresholdData[2];    /* Type of data used to threshold */
    swap2( &FieldHeader.iThresholdValue );
    swap2( &FieldHeader.iScale );
    //char  sEditCode[2];
    swap2( &FieldHeader.iPRT );                 /* PRT in microseconds */
    swap2( &FieldHeader.iBitsPerBin );          /* Must be 16 */
   // union uf_fsi2 Fsi;
}

printf("==========UF Field Header(%d) for %-2.2s=========₩n",sizeof(FieldHeader),
                                        DataHeader.Dsi[iMoment].sDataType );
printf("       Data position  = %d words₩n", FieldHeader.iDataPosition );
printf("        Scale factor  = %d₩n", FieldHeader.iScaleFactor );
printf("       Starting range = %5.3f km₩n",
           (float)FieldHeader.iStartRangeKm+(float)FieldHeader.iStartRangeMeters/1000.0 );
printf("           Bin spacing = %d meters₩n", FieldHeader.iBinSpacing );
printf("             Bin count = %d₩n", FieldHeader.iBinCount);
printf("           Pulse width = %d meters ₩n",  FieldHeader.iPulseWidth );
printf("    Horiz. beam width = %3.2f deg₩n", ((float)FieldHeader.iBeamWidthH)/64.0 );
printf(" Vertical beam width = %3.2f deg₩n", ((float)FieldHeader.iBeamWidthV)/64.0 );
printf("  Receiver bandwidth = %d MHz₩n", FieldHeader.iBandWidth );

switch(FieldHeader.iPolarization)
{
  case UF_POL_HORIZONTAL:
       printf( "          Polarization = Horizontal₩n" ); break;
  case UF_POL_VERTICAL:
       printf( "          Polarization = Vertical₩n" ); break;
  case UF_POL_CIRCULAR:
       printf( "          Polarization = Circular₩n" ); break;
  case UF_POL_ELLIPTICAL:
       printf( "          Polarization = Elliptical₩n" ); break;
  default:
       printf( "          Polarization = UNKNOWN (%d)₩n", FieldHeader.iPolarization); break;
}

printf("           Wavelength = %3.2f cm₩n", ((float)FieldHeader.iWaveLength)/64.0 );
printf("          Sample size = %d₩n", FieldHeader.iSampleSize );
printf("       Threshold data = %-2.2s₩n", FieldHeader.sThresholdData );
printf("      Threshold value = %d₩n", FieldHeader.iThresholdValue );
printf("                Scale = %d₩n", FieldHeader.iScale );
printf("            Edit code = %-2.2s₩n", FieldHeader.sEditCode );
printf("                 PRT = %d microseconds (%3.1f Hertz)₩n", FieldHeader.iPRT,
                                        1000000./(FLT8)FieldHeader.iPRT);
printf("         Bits per bin = %d₩n", FieldHeader.iBitsPerBin );


if (0 == strncmp( "VR", DataHeader.Dsi[iMoment].sDataType, 2 ))
{
   double fNyquist;

   printf("FieldHeader.iNyquistVelocity = %d₩n", FieldHeader.iNyquistVelocity );
   printf("FieldHeader.ipad1 = %d₩n", FieldHeader.ipad1 );
   printf("FieldHeader.iRadarConstant = %d₩n", FieldHeader.iRadarConstant );
   printf("FieldHeader.iNoisePower  = %d₩n", FieldHeader.iNoisePower  );
```

```
            printf("FieldHeader.iReceiverGain = %d₩n", FieldHeader.iReceiverGain );
            printf("FieldHeader.iPeakPower = %d₩n", FieldHeader.iPeakPower );
            printf("FieldHeader.iAntennaGain = %d₩n", FieldHeader.iAntennaGain);
            printf("FieldHeader.iPulseDuration = %d₩n", FieldHeader.iPulseDuration );

            fNyquist = ((1000000./(FLT8)FieldHeader.iPRT)*
                       ((float)FieldHeader.iWaveLength)/64.0)/(4.0*(float)FieldHeader.iScaleFactor);
            printf( "     Nyquist velocity = %4.2f m/s₩n", fNyquist );

         }

                  printf("₩n" );

         /*********************************************************************
         *         ReadField Data                                                *
         *********************************************************************/

         //int iBinCount;
         //short iDataArray[4200];
         iBinCount = FieldHeader.iBinCount;
         if (iBinCount>4200) iBinCount=4200;

         iReadSize = 2*iBinCount;

         iCount = gzread( stream, &iDataArray, iReadSize );
         if (iCount != iReadSize) perror( "fread" );
         iPosition += iReadSize;

         if (LITTLE_END)
         {
           swap2_array( iDataArray, iReadSize/2 );
         }

         printf( "===UF Field Data(%d) for %-2.2s====₩n", sizeof(iDataArray),
                    DataHeader.Dsi[iMoment].sDataType );

         strcpy(sThreshold, "  -.- " ) ;
         strcpy(sFormat, "%5.1f " ) ;
         fScale = (float)FieldHeader.iScaleFactor;

         printf( "%4d:", 0 );
         iTotalLength=5;

         for (iBin=0; iBin<FieldHeader.iBinCount; iBin++)
         {
            iValue = iDataArray[iBin];
            if( iValue == MaHeader.iNoDataValue )
               strcpy( sThisString, sThreshold ) ;
            else
               sprintf( sThisString, sFormat, ((FLT8)iValue)/fScale ) ;

            iThisLength = strlen(sThisString);

            if( iTotalLength+iThisLength > 100 )
            {
               printf( "₩n%4d:", iBin );
               iTotalLength=5;
            }

            printf( "%s", sThisString );
            iTotalLength += iThisLength;
         }

         printf( "₩n" ) ;

      } //===for iMoment

   SKIP_THIS_RAY:
      iSeekSize = iRecordLength + 4 - iPosition;
      if (iSeekSize>0) iCount = gzseek( stream, iSeekSize, SEEK_CUR );
      iCount = gzread( stream,&iRecordLength,  4 );
      iPosition = 0;
```

```
    } //==========while

  if (iCount != 0)
  {
     printf( "Bad read, got%d, wanted%d bytes₩n", iCount, (int)sizeof(MaHeader));
  }
  else
  {
     if (ferror(stream)) printf( "Read error.₩n" );
  }

  DONE:

  gzclose( stream );
  printf( "=== EXIT_SUCCESS. ====₩n" );
  exit( EXIT_SUCCESS );
} //=======main


/*  Byte swapping functions. */
static void swap2(void *word)
{
  unsigned char *byte;
  unsigned char temp;
  byte     = (unsigned char *)word;
  temp     = byte[0];
  byte[0] = byte[1];
  byte[1] = temp;
}

static void swap4(void *word)
{
  unsigned char *byte;
  unsigned char temp;
  byte     = (unsigned char *)word;
  temp     = byte[0];
  byte[0] = byte[3];
  byte[3] = temp;
  temp     = byte[1];
  byte[1] = byte[2];
  byte[2] = temp;
}

static void swap2_array
(short *buf,
 int iCount_a )
{
  short *end_addr;
  end_addr = buf +  iCount_a;
  while (buf < end_addr)
    {
      swap2(buf);
      buf++;
    }
  return;
}
```

제3장 CAPPI 자료생성

3.1 UF자료에서 CAPPI(1.5km)자료 생성

```
/*************************************************************
 * project : uf2txt                                          *
 * comfile : gcc -o  uf2txt uf2txt.c                    ₩    *
 *            -I/usr/local/trmm/GVBOX/include           ₩    *
 *            -L/usr/local/trmm/GVBOX/lib -lrsl -lm          *
 *************************************************************/
#include "rsl.h"
#include <string.h>
#include <stdlib.h>
#include <stdio.h>
#include <math.h>

#define XDIM 500        // X축 격자수
#define YDIM 500        // Y축 격자수
#define cell_size 1.0   // 1.0km

static char *RSL_ftype[] =
 {"DZ", "VR", "SW", "CZ", "ZT", "DR",
  "LR", "ZD", "DM", "RH", "PH", "XZ",
  "CD", "MZ", "MD", "ZE", "VE", "KD",
  "TI", "DX", "CH", "AH", "CV", "AV",
  "SQ","AT","BK"};

int main(int argc, char **argv)
{
  Radar  *radar;
  Volume *volume;
  Cappi  *cappi;
  Carpi  *carpi;
  int i, j;
  float ht;    //Cappi 고도
  float range; //관측거리 km
  float slon, slat;
  float dbz;
  float Z, R;

  radar = RSL_anyformat_to_radar(argv[1], NULL);

  slon  = (float)radar->h.lond + (float)radar->h.lonm/60 + (float)radar->h.lonm/3600;
  slat  = (float)radar->h.latd + (float)radar->h.latm/60 + (float)radar->h.latm/3600;

  ht = (1500.0 - radar->h.height)/1000.; //1500-싸이트해발고도
  range = (radar->v[CZ_INDEX]->sweep[0]->ray[0]->h.nbins *
           radar->v[CZ_INDEX]->sweep[0]->ray[0]->h.gate_size +
           radar->v[CZ_INDEX]->sweep[0]->ray[0]->h.range_bin1)/1000;

  for (i=0;i<radar->h.nvolumes;i++)
  {
     if (radar->v[i] != NULL)
     {
        printf(" radar->v[%d] = %s ₩n",i, RSL_ftype[i]); //field type
        for (j=0;j <radar->v[i]->h.nsweeps;j++)
        {
          // sweep Elevation
          printf(" radar->v[%d]->sweep[%d]=%5.2f, rays=%d ₩n",i, j,
                   radar->v[i]->sweep[j]->h.elev,radar->v[i]->sweep[j]->h.nrays);
        }
     }
  }
```

```
  cappi = RSL_cappi_at_h( radar->v[CZ_INDEX], ht, range);
  cappi->sweep = RSL_copy_sweep(radar->v[CZ_INDEX]->sweep[0]);
  carpi = RSL_cappi_to_carpi(cappi,cell_size,cell_size,cappi->lat,cappi->lon,XDIM,YDIM,XDIM/2,YDIM/2);

  for(j=0;j<YDIM;j++)
  {
        for(i=0;i<XDIM;i++)
        {
            dbz= carpi->f(carpi->data[j][i]);
            if (dbz == 131072.0) dbz =-99.9; //bad & noecho
            Z = pow(10.0, dbz/10.0);
            R = pow(Z/200.0, 1.0/1.6);       // Z = 200 * R 1.6
            if(dbz > -32.0 && dbz < 100.0)
               printf("[%03d][%03d] = %6.2f dBz %6.2f mm/hr Wn", j, i, dbz,R);
        }
  }
  RSL_free_carpi(carpi);
  RSL_free_cappi(cappi);
  RSL_free_radar(radar);
}
```

3.1.1 컴파일 및 결과

```
[operator@gng-disp1 uf2txt]$ ./uf2txt ./data/RDR_PSN_201012031000.uf|more
 radar->v[0] = DZ
 radar->v[0]->sweep[0]= 0.00, rays=360
 radar->v[0]->sweep[1]= 0.39, rays=360
 radar->v[0]->sweep[2]= 0.78, rays=360
 radar->v[0]->sweep[3]= 1.19, rays=360
 radar->v[0]->sweep[4]= 1.59, rays=360
 radar->v[0]->sweep[5]= 1.98, rays=360
 radar->v[0]->sweep[6]= 2.98, rays=360
 radar->v[0]->sweep[7]= 4.19, rays=360
 radar->v[0]->sweep[8]= 5.69, rays=360
 radar->v[0]->sweep[9]= 7.48, rays=360
 radar->v[0]->sweep[10]= 9.80, rays=360
 radar->v[0]->sweep[11]=12.48, rays=360
 radar->v[0]->sweep[12]=15.80, rays=360
 radar->v[1] = VR
 radar->v[1]->sweep[0]= 0.00, rays=360
 radar->v[1]->sweep[1]= 0.39, rays=360
 radar->v[1]->sweep[2]= 0.78, rays=360
 radar->v[1]->sweep[3]= 1.19, rays=360
 radar->v[1]->sweep[4]= 1.59, rays=360
 radar->v[1]->sweep[5]= 1.98, rays=360
  ~~~~
[034][163] =   7.00 dBz   0.10 mm/hr
[034][164] =   7.00 dBz   0.10 mm/hr
[035][161] =   7.00 dBz   0.10 mm/hr
[035][162] =   6.50 dBz   0.09 mm/hr
[038][285] =   6.00 dBz   0.09 mm/hr
[038][286] =   6.00 dBz   0.09 mm/hr
[039][286] =   5.50 dBz   0.08 mm/hr
[039][287] =   5.50 dBz   0.08 mm/hr
[039][288] =   6.00 dBz   0.09 mm/hr
[039][364] =   7.00 dBz   0.10 mm/hr
[039][365] =   7.00 dBz   0.10 mm/hr
[040][366] =   7.00 dBz   0.10 mm/hr
[040][367] =   7.00 dBz   0.10 mm/hr
[047][259] =   5.50 dBz   0.08 mm/hr
[047][260] =   5.50 dBz   0.08 mm/hr
```

3.2 UF자료에서 임의 위·경도자료 생성

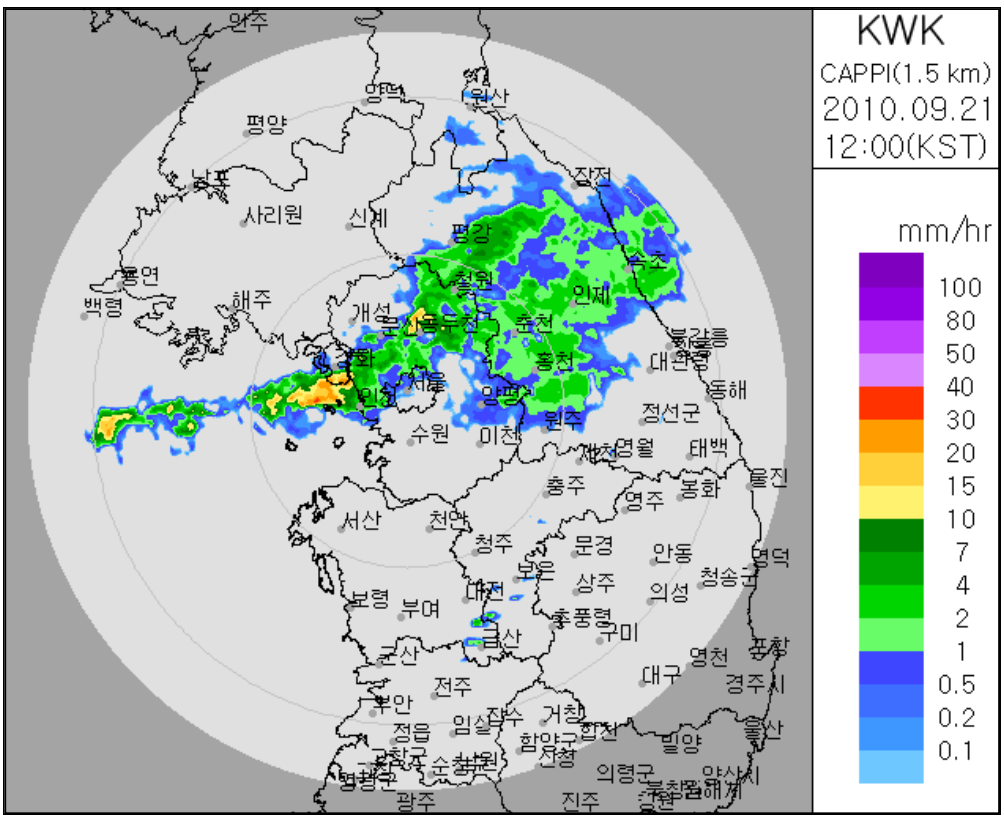
```
/*****************************************************************
 * project : uf2point                                            *
 * comfile : gcc -o  uf2point uf2pont.c ../gis/azedproj.c  ₩     *
 *            -I/usr/local/trmm/GVBOX/include              ₩     *
 *            -L/usr/local/trmm/GVBOX/lib -lrsl -lm              *
 *****************************************************************/
#include "rsl.h"
#include <string.h>
#include <stdlib.h>
#include <stdio.h>
#include <math.h>
#include "../gis/map_ini.h"

#define XDIM 500        // X축 격자수
#define YDIM 500        // Y축 격자수
#define cell_size 1.0   // 1.0km
int main(int argc, char **argv)
{
  Radar  *radar;
  Volume *volume;
  Cappi  *cappi;
  Carpi  *carpi;
  int i, j;
  float ht;    //Cappi 고도
  float range; //관측거리 km
  float dbz;
  float Z, R;

  float  x2, y2;
  float  alon, alat;
  struct azed_parameter  map;

  map.Re     = 6371.00877;  //지구반경
  map.grid   = 1.0;         //Cell size
  //map.slon = 128.52;      //위,경도를 uf자료에서 가져옴
  //map.slat = 35.715;
  //map.olon = 128.52;
  //map.olat = 35.715;
  map.xo     = (float)XDIM/2.0;  // x축 기준점
  map.yo     = (float)YDIM/2.0;  // y축 기준점
  map.first  = 0;

  radar = RSL_anyformat_to_radar(argv[1], NULL);

  map.slon  = (float)radar->h.lond + (float)radar->h.lonm/60 + (float)radar->h.lons/3600;
  map.slat  = (float)radar->h.latd + (float)radar->h.latm/60 + (float)radar->h.lats/3600;
  map.olon  = map.slon;
  map.olat  = map.slat;

  ht = (1500.0 - radar->h.height)/1000.; //1500-싸이트해발고도
  range = (radar->v[CZ_INDEX]->sweep[0]->ray[0]->h.nbins *
           radar->v[CZ_INDEX]->sweep[0]->ray[0]->h.gate_size +
           radar->v[CZ_INDEX]->sweep[0]->ray[0]->h.range_bin1)/1000;

  cappi = RSL_cappi_at_h( radar->v[CZ_INDEX], ht, range);
  carpi = RSL_cappi_to_carpi(cappi,cell_size,cell_size,cappi->lat,cappi->lon,XDIM,YDIM,XDIM/2,YDIM/2);

  alon = atof(argv[2]);
  alat = atof(argv[3]);
  azedproj(&alon, &alat, &x2, &y2, 0, map);

  int yy,xx;
```

```
  yy = (int)y2;
  xx = (int)x2;
  dbz= carpi->f(carpi->data[yy][xx]);
  if (dbz == 131072.0) dbz =-99.9; //bad & noecho
  Z = pow(10.0, dbz/10.0);
  R = pow(Z/200.0, 1.0/1.6);        // Z = 200 * R 1.6
  if(dbz > -32.0 && dbz < 100.0)
     printf("[%03d][%03d] = %6.2f dBz %6.2f mm/hr ₩n", yy, xx, dbz,R);
  else
     printf("[%03d][%03d] = bad & noecho ₩n", yy, xx);

  RSL_free_carpi(carpi);
  RSL_free_cappi(cappi);
  RSL_free_radar(radar);
}
```

o. 실행결과

```
radar@radar3:/RDPS/KWG/uf2point$ ./uf2point ./data/RDR_KWK_QCD_201009211200.uf 126.5 37.5
 [256][208] =  42.50 dBz  16.52 mm/hr
```

3.3 UF자료에서 CAPPI 영상 생성



o. 컴파일(Makefile)

```
map_lib = ../gis/azedproj.c
INCLUDEDIRS=-I.  -I/usr/local/trmm/GVBOX/include
LIBDIRS=-L.  -L/usr/local/trmm/GVBOX/lib
LIBS=-lrsl -lgd -lpng -lz -lfreetype -lfl

uf2png: uf2png.c
        gcc $(INCLUDEDIRS) -o ./uf2png uf2png.c $(map_lib) $(LIBDIRS) $(LIBS)
```

```
//uf2png.c
#include <string.h>
#include <stdlib.h>
#include <gd.h>
#include <math.h>
#include <gdfontt.h>
#include <gdfonts.h>
#include <gdfontmb.h>
#include <gdfontl.h>
#include <gdfontg.h>
#include <zlib.h>
#include <zconf.h>
#include "rsl.h"
#include "../gis/map_ini.h"

#define XDIM 600
#define YDIM 600

int main(int argc, char *argv[])
{
  gdImagePtr im;
  FILE *pngout;
  int color[255];
  int i, j, k;
  char infile[90];
  char outfile[90];

  float  x2, y2;
  float  alon, alat;
  struct azed_parameter  map;
  map.Re    = 6371.00877;
  map.grid = 1.0;          //Cell size
  map.xo    = (float)XDIM/2.0;  // x. ...
  map.yo    = (float)YDIM/2.0;  // y. ...
  map.first= 0;

 if ( argc != 3 )
 {  printf(" usage ./UfToSitePng uffilename height ₩n");
    exit(0);
 }

 im = gdImageCreate(XDIM+120, YDIM);  //color table area 120 pixel

        color[0]  = gdImageColorAllocate(im, 224, 224, 224);
        color[1]  = gdImageColorAllocate(im,  62, 148, 255);
```

```
        color[2]  = gdImageColorAllocate(im,  62, 116, 255);
        color[3]  = gdImageColorAllocate(im,  40,  49, 255);
        color[4]  = gdImageColorAllocate(im,  70, 255,  70);
        color[5]  = gdImageColorAllocate(im,   0, 224,  80);
        color[6]  = gdImageColorAllocate(im,   0, 175, 101);
        color[7]  = gdImageColorAllocate(im,   0, 150, 101);
        color[8]  = gdImageColorAllocate(im, 255, 229,   0);
        color[9]  = gdImageColorAllocate(im, 255, 190,   0);
        color[10] = gdImageColorAllocate(im, 255, 112,   0);
        color[11] = gdImageColorAllocate(im, 215,   0,   0);
        color[12] = gdImageColorAllocate(im, 255, 206, 255);
        color[13] = gdImageColorAllocate(im, 255, 170, 255);
        color[14] = gdImageColorAllocate(im, 255, 125, 250);
        color[15] = gdImageColorAllocate(im, 255,   0, 255);
        color[20] = gdImageColorAllocate(im,   0,   0,   0);
        color[21] = gdImageColorAllocate(im, 255, 255, 255);
        color[22] = gdImageColorAllocate(im, 224, 224, 224);
        color[23] = gdImageColorAllocate(im, 196, 196, 196);

  gdImageFilledRectangle(im, 0, 0, XDIM, YDIM, color[23]);

/*********************************************************
 *          UF를 CAPPI로 읽어 그리기                     *
 *********************************************************/
 Radar  *radar;
 Volume *volume;
 Cappi  *cappi;
 Carpi  *carpi;
 int range;
 float ht;

 radar = RSL_anyformat_to_radar(argv[1], NULL);
 volume = radar -> v[CZ_INDEX];

 printf("date = %04d.%02d.%02d ₩n",radar->h.year,radar->h.month,radar->h.day);
 printf("time = %02d:%02d:%02d ₩n",radar->h.hour,radar->h.minute,(int)radar->h.sec);
 printf("site = %s ₩n",radar->h.name);
 printf("name = %s ₩n",radar->h.radar_name);
 printf("type = %s ₩n",radar->h.radar_type);
 printf("lat  = %d.%d.%d ₩n",radar->h.latd,radar->h.latm,radar->h.latm);
 printf("lon  = %d.%d.%d ₩n",radar->h.lond,radar->h.lonm,radar->h.lonm);
 printf("height = %d m₩n",radar->h.height);
 map.slon  = (float)radar->h.lond + (float)radar->h.lonm/60 + (float)radar->h.lonm/3600;
 map.slat  = (float)radar->h.latd + (float)radar->h.latm/60 + (float)radar->h.latm/3600;
 map.olon  = map.slon;
 map.olat  = map.slat;
 range = (volume->sweep[0]->ray[0]->h.nbins * volume->sweep[0]->ray[0]->h.gate_size +
          volume->sweep[0]->ray[0]->h.range_bin1)/1000;
 printf("range = %d km ₩n",range);

 ht=(atof(argv[2])*1000.0-(float)radar->h.height)/1000.0;
 cappi = RSL_cappi_at_h( radar->v[CZ_INDEX], ht, (float)XDIM/2);
 carpi = RSL_cappi_to_carpi(cappi , map.grid, map.grid,cappi->lat,cappi->lon,XDIM,YDIM,XDIM/2,YDIM/2);

 float A;
 char cindex;

 for(j=0;j<YDIM;j++)
 {
```

```
        for(i=0;i<XDIM;i++)
        {
                A = carpi->f(carpi->data[j][i]);
                if (A != 131072)
                {
                   switch((char)A)
                   {
                   case 1 ... 13:
                       cindex = 1;
                            break;
                   case 14 ... 18:
                       cindex = 2;
                            break;
                   case 19 ... 23:
                       cindex = 3;
                            break;
                   case 24 ... 28:
                       cindex = 4;
                            break;
                   case 29 ... 34:
                       cindex = 5;
                            break;
                   case 35 ... 39:
                       cindex = 6;
                            break;
                   case 40 ... 42:
                       cindex = 7;
                            break;
                   case 43 ... 44:
                       cindex = 8;
                            break;
                   case 45:
                       cindex = 9;
                            break;
                   case 46 ... 47:
                       cindex = 10;
                             break;
                   case 48 ... 49:
                       cindex = 11;
                             break;
                   case 50 ... 51:
                       cindex = 12;
                             break;
                   case 52 ... 54:
                       cindex = 13;
                             break;
                   case 55:
                       cindex = 14;
                             break;
                   case 56 ... 80:
                       cindex = 15;
                             break;
                   default:
                       cindex = 0;
                   }
                   gdImageSetPixel(im, i , YDIM-j, color[cindex]);
                }
                else
                {
```

```
                    gdImageSetPixel(im, i , YDIM-j, color[0]);
                }
                int rdasi =(int)sqrt((XDIM/2-i)*(XDIM/2-i)+ (YDIM/2-j)*(YDIM/2-j));
                if (rdasi > range) gdImageSetPixel(im, i , YDIM-j, color[23]);

        }
  }

/*********************************************************
 *          관측반경 그리기                                *
 *********************************************************/
  float zoom;
  zoom = 2.0 * map.grid;
  gdImageArc(im, XDIM/2, YDIM/2, 100*(int)zoom, 100*(int)zoom, 0, 360, color[20]);
  gdImageArc(im, XDIM/2, YDIM/2, 200*(int)zoom, 200*(int)zoom, 0, 360, color[20]);
  gdImageArc(im, XDIM/2, YDIM/2,        2*range,        2*range, 0, 360, color[20]);
  gdImageRectangle(im, 0, 0, XDIM, YDIM-1, color[21]);

/*********************************************************
 *          color table 그리기                             *
 *********************************************************/
  char str[255];
  char *f = "../font/gulim.ttc";
  double sz = 13.;
  int brect[8];
  char *err;

  gdImageFilledRectangle(im, XDIM, 0, XDIM+ 120-1, YDIM-1, color[21]);
  gdImageRectangle(im, XDIM,        0, XDIM+ 120-1, YDIM-1, color[20]);
  gdImageRectangle(im, XDIM, YDIM/4, XDIM+ 120-1, YDIM-1, color[20]);

  static char  rain[][15] = {"0.0","0.2","0.5","1","2","5","10","15",
                             "20","25","30","40","50","80","100","150"};

  int xoff, xwith;
  int yoff, ywith, ydiv;
  for (i=1;i<16 ;i+ + )
  {
     xoff  = 30;            // ----->30
     xwith = xoff +  30;    // ----->30 <--30-->
     yoff  = 20;            // ----->20
     ydiv  = YDIM * 3 / (4 * 17);
     ywith = i*ydiv+ yoff;    // ----->20 <-- i * 20 -->
     gdImageFilledRectangle(im, XDIM+ xoff, YDIM-ywith, XDIM+ xwith, YDIM-ywith+ ydiv, color[i]);
     sprintf(str, " %s",rain[i]);
     gdImageStringFT(im, &brect[0], color[20], f, 12.0, 0.0,XDIM+ xwith+ 5 , YDIM-ywith+ 15, str);
  }

  gdImageStringFT(im, &brect[0], color[20], f, 20.0, 0.0,XDIM+ 17 , yoff*2 ,"CAPPI");

  sprintf(str,"%s km",argv[2]);
  gdImageStringFT(im, &brect[0], color[20], f, 14.0, 0.0,XDIM+ 28 , yoff*3 ,str);

  sprintf(str,"%04d.%02d.%02d",radar->h.year,radar->h.month,radar->h.day);
  gdImageStringFT(im, &brect[0], color[20], f, 15.0, 0.0,XDIM+ 14 , yoff*5 ,str);
  sprintf(str,"%02d:%02d:%02d",radar->h.hour,radar->h.minute,(int)radar->h.sec);
  gdImageStringFT(im,&brect[0], color[20], f, 15.0, 0.0,XDIM+ 20 , yoff*6 ,str);
  gdImageStringFT(im,&brect[0], color[20], f, 12.0, 0.0,XDIM+ xwith-15 , YDIM-ywith-8,"mm/hr");
```

```
/*********************************************************
 *          AWS 지명 그리기                                *
 *********************************************************/
  map.first = 0;

  FILE *fpAws;
  int num_aws;
  char aws_name[50];;
  float aws_lat, aws_lon;

  fpAws = fopen("../aws/aws_pos_han_2.txt","r");
  fscanf(fpAws,"%d",&num_aws);

  for (i = 0; i < num_aws ; i++)
  {
        fscanf(fpAws,"%s %f %f",aws_name,&aws_lat,&aws_lon);
        alon=aws_lon;
        alat=aws_lat;
        azedproj(&alon, &alat, &x2, &y2, 0, map);
        if (x2 > 0 && x2 < XDIM && y2 > 0 && y2 < YDIM)
           gdImageStringFT(im,&brect[0],color[20],f,10.0, 0.0,(int)x2,YDIM-(int)y2, aws_name);
  }
  fclose(fpAws);

/*********************************************************
 *          지도 그리기                                   *
 *********************************************************/
  FILE *fpMap;
  int line, type;
  map.first = 0;

  fpMap = fopen ("../gis/fine_kr.dat" , "r");

  float _x=0, _y=0;

  while (fgets(str, 1024, fpMap) != NULL)
  {
     j = 0;
     sscanf(str, "%d %d", &line, &type);

     fgets(str, 1024, fpMap);
     sscanf(str, "%f %f", &aws_lon, &aws_lat);
     alon=aws_lon;
     alat=aws_lat;
     azedproj(&alon, &alat, &x2, &y2, 0, map);
     _x = x2;
     _y = y2;
     for (i=0; i<line - 1; i++)
     {
        fgets(str, 1024, fpMap);

        if (type != 3)
        {
          sscanf(str, "%f %f", &aws_lon, &aws_lat);
          alon=aws_lon;
          alat=aws_lat;
          azedproj(&alon, &alat, &x2, &y2, 0, map);
          if (x2 > 0 && x2 < XDIM && y2 > 0 && y2 < YDIM)
          {
```

```
                gdImageLine(im, (int)(x2+0.5), YDIM-(int)(y2+0.5)-1, (int)(_x+0.5),
                                                YDIM-(int)(_y+0.5)-1, color[20]);
            }
            _x = x2;
            _y = y2;
          }
        }
    }
    fclose(fpMap);

  /*********************************************************
    *    이미지를 화일로 저장                                *
    *********************************************************/
    strncpy(infile, argv[1], strlen(argv[1])-strlen(strrchr(argv[1], '.')));
    sprintf(outfile, "%s.png",infile);
    printf("Outfile   : %s ₩n",outfile);
    pngout   = fopen(outfile,"wb");
    gdImagePng(im,pngout);
    fclose(pngout);
    gdImageDestroy(im);
}
```

제4장 합성자료

4.1 CAPPI(1.5km)합성자료

4.1.1 파일명 : c1cm_년월일시분.bin.gz

4.1.2 자료구조

- o 960 * 1200 * 1bytes(1km격자)
- o echo : 0 ~ 100(dBz)
- o 관측반경 밖 : -128
- o no echo : -127

4.1.3 지도 좌표 형식

- o 지도형식 : Lambert conformal conic projection
- o 격자간격 : 1km
- o 표준위도 : 30~60
- o 기준점위경도 : 126/38
- o 기준점X,Y좌표: 400/789

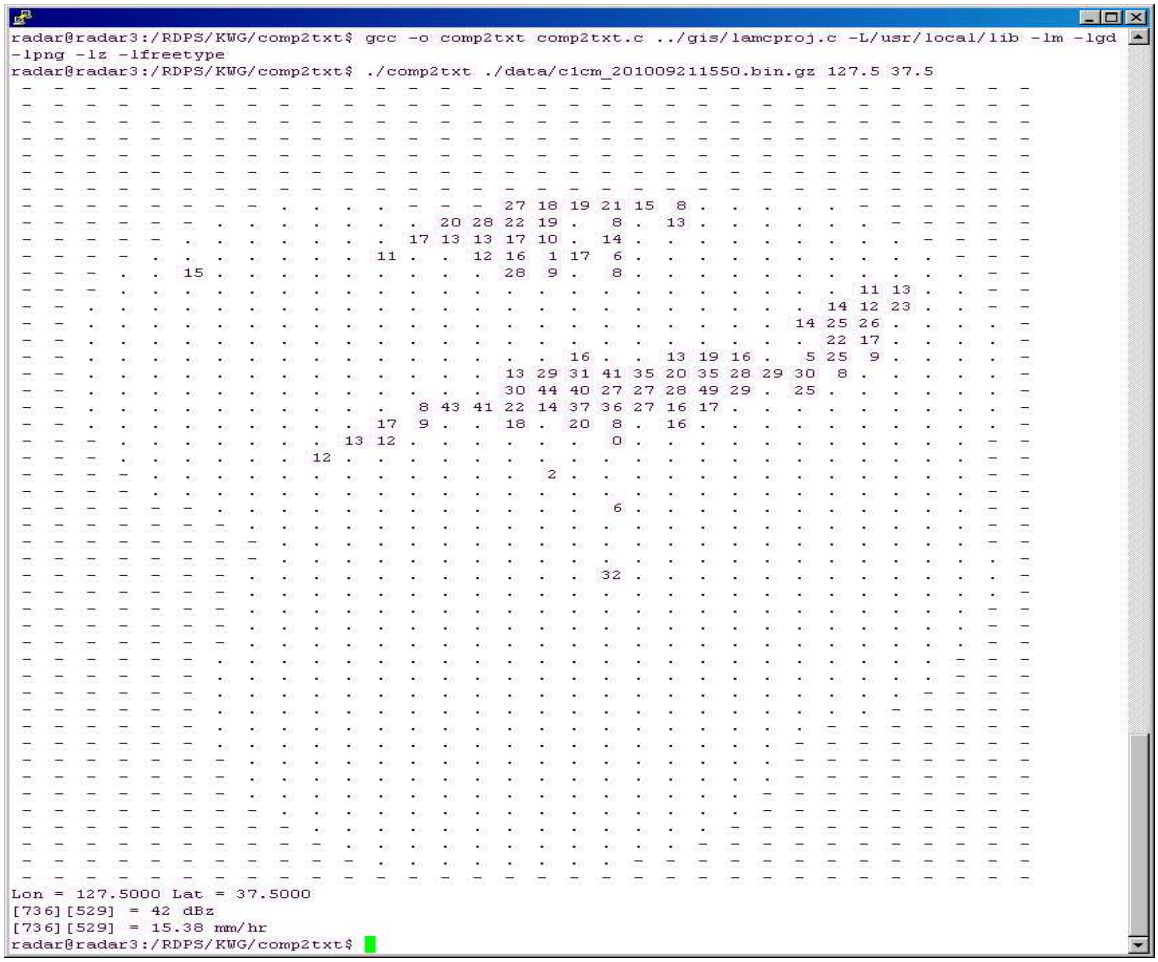
4.1.4 자료변환 예시

- o 합성 이진자료를 텍스트로 변환

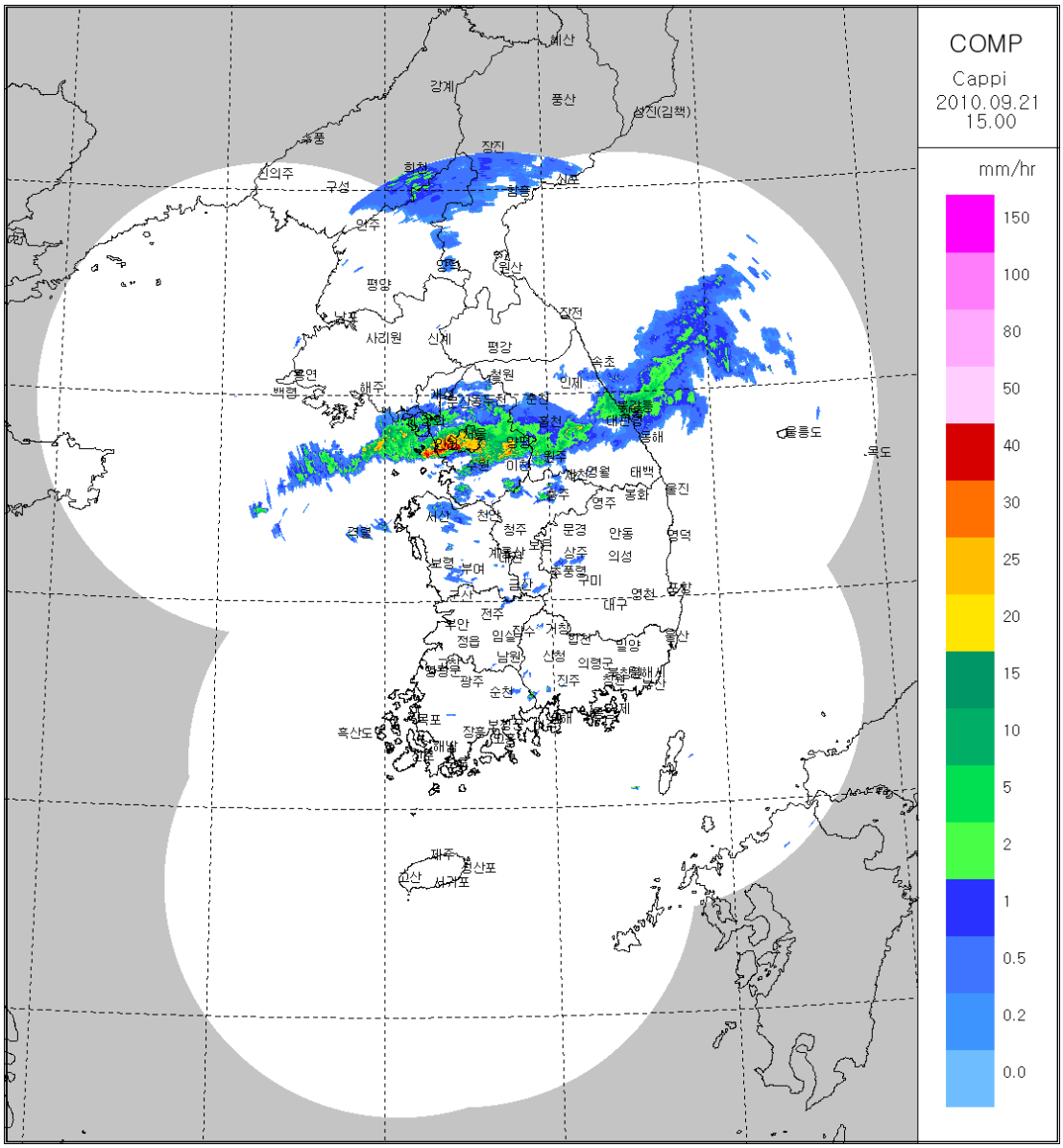
```
/*********************************************************
**  gcc -o comp2txt comp2txt.c ../gis/lamcproj.c          **
**  -L/usr/local/lib -lm -lgd -lpng -lz -lfreetype        **
*********************************************************/
#include <string.h>
#include <stdlib.h>
#include <stdio.h>
#include <math.h>
#include <zlib.h>
#include <zconf.h>
#include "../gis/map_ini.h"
#define  XDIM  960
#define  YDIM  1200
char DATA[YDIM][XDIM];

int main(int argc, char *argv[])
{
  FILE  *fp1;
  int i, j, k;
  float  x2, y2;
  float  alon, alat;
  struct lamc_parameter map;
  float dbz, z, rain;
```

```
  map.Re    = 6370.19584; // 사용할 지구반경 [ km ]
  map.grid  = 1.0;        // 격자간격       [ km ]
  map.slat1 = 30.0;       // 표준위도       [degree]
  map.slat2 = 60.0;       // 표준위도       [degree]
  map.olat  = 38.0;       // 기준점의 경도  [degree]
  map.olon  = 126.0;      // 기준점의 위도  [degree]
  map.xo    = 400;        // 기준점의 X좌표 [격자거리]
  map.yo    = 789;        // 기준점의 Y좌표 [격자거리]
  map.first = 0;          // 시작여부 (0 = 시작)

  fp1 = gzopen(argv[1],"rb");
  gzread(fp1, &DATA, sizeof(DATA));

  for (j=1;j<YDIM;j+=25)
  {
     for (i=0;i<XDIM;i+=30)
     {
        if(DATA[YDIM-j][i] == -127)
        {
                printf(" . ");
        }
        else if(DATA[YDIM-j][i] == -128)
        {
                printf(" - ");
        }
        else
        {
                printf(" %2d",DATA[YDIM-j][i]);
        }
     }
     printf("₩n");
  }
  gzclose(fp1);

  //해당 위경도의 레이더 값 표출
  alon=atof(argv[2]); //aws_lon;
  alat=atof(argv[3]); //aws_lat;
  lamcproj(&alon, &alat, &x2, &y2, 0, map);
  printf("Lon = %7.4f Lat = %7.4f ₩n",alon, alat);

  if (DATA[(int)y2][(int)x2] > 0 && DATA[(int)y2][(int)x2] < 100 )
  {
          printf("[%03d][%03d] = %d dBz ₩n",(int)y2,(int)x2, DATA[(int)y2][(int)x2]);

          dbz = (float)DATA[(int)y2][(int)x2];
          z = pow(10.0, dbz/10.0);
          rain = pow(z/200.0, 1.0/1.6);

          printf("[%03d][%03d] = %5.2f mm/hr ₩n",(int)y2, (int)x2, rain);
  }
  else
  {
          printf("No echo or Vad vaild ₩n");
  }
}
```

* 위경도 126.5, 37.5의 레이더 값은 42dBz(15.38mm/hr)


radar@radar3:/RDPS/KWG/comp2txt$ gcc -o comp2txt comp2txt.c ../gis/lamcproj.c -L/usr/local/lib -lm -lgd
-lpng -lz -lfreetype
radar@radar3:/RDPS/KWG/comp2txt$ ./comp2txt ./data/c1cm_201009211550.bin.gz 127.5 37.5
Lon = 127.5000 Lat = 37.5000
[736][529] = 42 dBz
[736][529] = 15.38 mm/hr
radar@radar3:/RDPS/KWG/comp2txt$

o 합성 이진자료를 영상으로 변환



[그림2 영상으로 표출]

```
/************************************************************
 ** project : comp2png                                      **
 ** comfile :                                               **
 ** gcc -o comp2png comp2png.c ../gis/lamcproj.c     ₩     **
 ** -L/usr/local/lib -lm -lgd -lpng -lz -lfreetype          **
 ************************************************************/
#include <string.h>
#include <stdio.h>
#include <stdlib.h>
#include <gd.h>
#include <math.h>
```

```
#include <gdfontt.h>
#include <gdfonts.h>
#include <gdfontmb.h>
#include <gdfontl.h>
#include <gdfontg.h>
#include <zlib.h>
#include <zconf.h>
#include <malloc.h>
#include "../gis/map_ini.h"

#define  XDIM  960
#define  YDIM  1200

int date_time( char *filename );
int read_color(gdImagePtr im, int color[]);
int read_radar(gdImagePtr im, int color[], char *filename);
int latlon_draw( gdImagePtr im, int color[]);
int title_draw( gdImagePtr im, int color[]);
int aws_draw( gdImagePtr im, int color[]);
int map_draw( gdImagePtr im, int color[]);

struct lamc_parameter map;
float alon, alat, x2, y2;
int YY, MM, DD, HH, MI;

int main(int argc, char *argv[])
{
  gdImagePtr im;
  FILE *pngout, *jpegout;
  int color[100];
  int i, j, k;
  char infile[100]="";
  char outfile[100]="";
  char tmp[100]="";

  map.Re      = 6370.19584;  // 사용할 지구반경 [ km ]
  map.grid    = 1.0;         // 격자간격        [ km ]
  map.slat1   = 30.0;        // 표준위도        [degree]
  map.slat2   = 60.0;        // 표준위도        [degree]
  map.olat    = 38.0;        // 기준점의 경도   [degree]
  map.olon    = 126.0;       // 기준점의 위도   [degree]
  map.xo      = 400;         // 기준점의 X좌표  [격자거리]
  map.yo      = 789;         // 기준점의 Y좌표  [격자거리]
  map.first   = 0;           // 시작여부 (0 = 시작)

  sprintf(tmp, "%s",argv[1]);
  date_time(tmp);

  im = gdImageCreate(XDIM+150, YDIM);//이미지 생성 title box여백 150
  read_color(im, color);              //cololr load
  gdImageFilledRectangle(im, 0, 0, XDIM, YDIM, color[19]);  //echo box 배경색
  read_radar( im, color, argv[1]);   //레이더자료 읽어 그리기
  latlon_draw( im, color );          //위,경도 그리기
  gdImageFilledRectangle(im, XDIM, 0, XDIM+150, YDIM, color[21]);  //title box 배경색
```

```
  title_draw( im, color );              //title box 그리기
  aws_draw( im, color );                  //aws지명 그리기
  map_draw( im, color );                  //지도 그리기

  gdImageRectangle(im,     0,    0,         XDIM, YDIM-1, color[22]); // echo box 테두리선
  gdImageRectangle(im, XDIM,    0, XDIM+150-1, YDIM-1, color[22]);  // title box 테두리선
  gdImageRectangle(im, XDIM, 150, XDIM+150-1, YDIM-1, color[22]);  // title box 중간선
  //=============================================================
  strncpy(infile, argv[1], strlen(argv[1])-strlen(strrchr(argv[1], '.')));
  sprintf(outfile, "%s.png",infile);
  printf("Outfile  : %s ₩n",outfile);
  pngout  = fopen(outfile,"wb");
  gdImagePng(im,pngout);
  fclose(pngout);
  gdImageDestroy(im);
}
//===============main end ============================

/*********************************************************
 *          레이더 자료 읽어 그리기                                  *
 *********************************************************/
int read_radar(gdImagePtr im, int color[], char *filename)
{
  FILE    *fpRdr;
  int i, j;
  int n, cindex;
  char  buf[1200][960];

   fpRdr = gzopen(filename,"rb");

   gzread(fpRdr, &buf, sizeof(buf));

   for (j=0;j<1200-1;j++)
   {
     for (i=1;i<960;i++)
     {
                        switch(buf[j][i])
                        {
                                case 1 ... 11:
                                   cindex = 1;
                                   break;
                                case 12 ... 18:
                                   cindex = 2;
                                   break;
                                case 19 ... 23:
                                   cindex = 3;
                                   break;
                                case 24 ... 28:
                                   cindex = 4;
                                   break;
                                case 29 ... 34:
                                   cindex = 5;
                                   break;
                                case 35 ... 39:
```

```
                                        cindex = 6;
                                        break;
                                case 40 ... 42:
                                        cindex = 7;
                                        break;
                                case 43 ... 44:
                                        cindex = 8;
                                        break;
                                case 45:
                                        cindex = 9;
                                        break;
                                case 46 ... 47:
                                        cindex = 10;
                                        break;
                                case 48 ... 49:
                                        cindex = 11;
                                        break;
                                case 50:
                                        cindex = 12;
                                        break;
                                case 51 ... 54:
                                        cindex = 13;
                                        break;
                                case  55:
                                        cindex = 14;
                                        break;
                                case 56 ... 58:
                                        cindex = 15;
                                        break;
                                case 59 ... 99:
                                        cindex = 16;
                                        break;

                                case -128 :
                                        cindex = 20; //.... ..
                                        break;
                                case -127 :
                                        cindex = 19; //no echo
                                        break;
                                default:
                                        cindex = 19;
                        }
                gdImageSetPixel(im, i, YDIM-j, color[cindex]);
        }
  }
  gzclose(fpRdr);

 return 0;
}

/*********************************************************
 *    color table load                                   *
 *********************************************************/
int read_color(gdImagePtr im, int         color[])
```

```
{
        color[0] = gdImageColorAllocate(im, 110, 190, 255);
        color[1] = gdImageColorAllocate(im, 62, 148, 255);
        color[2] = gdImageColorAllocate(im, 62, 116, 255);
        color[3] = gdImageColorAllocate(im, 40, 49, 255);
        color[4] = gdImageColorAllocate(im, 70, 255, 70);
        color[5] = gdImageColorAllocate(im, 0, 224, 80);
        color[6]= gdImageColorAllocate(im, 0, 175, 101);
        color[7] = gdImageColorAllocate(im, 0, 150, 101);
        color[8] = gdImageColorAllocate(im, 255, 229, 0);
        color[9] = gdImageColorAllocate(im, 255, 190, 0);
        color[10] = gdImageColorAllocate(im, 255, 112, 0);
        color[11] = gdImageColorAllocate(im, 215, 0, 0);
        color[12] = gdImageColorAllocate(im, 255, 206, 255);
        color[13] = gdImageColorAllocate(im, 255, 170, 255);
        color[14] = gdImageColorAllocate(im, 255, 125, 250);
        color[15] = gdImageColorAllocate(im, 255,  0, 255);
        color[16] = gdImageColorAllocate(im, 136, 136, 136);
        color[17] = gdImageColorAllocate(im, 163, 163, 163);
        color[18] = gdImageColorAllocate(im, 110, 190, 255);

        color[19] = gdImageColorAllocate(im, 224, 224, 224); //no echo
        color[20] = gdImageColorAllocate(im, 196, 196, 196); //... ..
        color[21] = gdImageColorAllocate(im, 255, 255, 255);
        color[22] = gdImageColorAllocate(im, 0, 0, 0);


        return 0;
}

/*********************************************
 * 경도, 위도선 그리기                       *
 *********************************************/
int latlon_draw( gdImagePtr im, int color[])
{
    int     lon, lat;
    float   x1, y1;

   // 경도선 /
   for (lon = 120; lon < 140; lon += 2)
    {
        alon = (float)lon;
        alat = 80;
        lamcproj(&alon, &alat, &x1, &y1, 0, map);

        alon = (float)lon;
        alat = 0;
        lamcproj(&alon, &alat, &x2, &y2, 0, map);
        gdImageDashedLine(im, (int)x1, YDIM-(int)y1, (int)x2, YDIM-(int)y2, color[22]);
    }

   // 위도선 /
    for (lat = 30; lat < 50; lat += 2)
    {
```

```
        for (lon = 120*10; lon < 140*10; lon += 1)
        {
            alon = (float)lon*0.1;
            alat = (float)lat;
            lamcproj(&alon, &alat, &x1, &y1, 0, map);

            alon = (float)lon*0.1 +  1*0.05;
            alat = (float)lat;
            lamcproj(&alon, &alat, &x2, &y2, 0, map);
            gdImageLine(im, (int)x1, YDIM-(int)y1, (int)x2, YDIM-(int)y2, color[22]);
        }
    }

  return 0;
}

/*********************************************************
*           color table 그리기                                    *
 *********************************************************/
int title_draw( gdImagePtr im, int color[])
{
  char str[255];
  char *f = "../font/gulim.ttc";
  double sz = 13.;
  int brect[8];
  char *err;
  int i;

  sprintf(str, " %s","COMP");
  err = gdImageStringTTF(im,&brect[0],color[22], f, 20.0,0.0,XDIM+25 , 50, str);

  sprintf(str, " %s","Cappi");
  gdImageStringFT(im, &brect[0], color[22], f, 16.0, 0.0,XDIM+30 , 85, str);

  sprintf(str, "%04d.%02d.%02d", YY,MM,DD);
  gdImageStringFT(im, &brect[0], color[22], f, 16.0, 0.0,XDIM+20 , 110, str);
  sprintf(str, "%02d.%02d", HH, MI);
  gdImageStringFT(im, &brect[0], color[22], f, 16.0, 0.0,XDIM+50 , 130, str);

  static                                char                                                                    rain[][15]
={"0.0","0.2","0.5","1","2","5","10","15","20","25","30","40","50","80","100","150"};
  int xoff, xwith; //x. .., .
  int yoff, ywith; //y. .., .
  for (i=0;i<16 ;i++ )
  {
       xoff = 30;
       xwith = 80; //80-30=50
       ywith = 60; //40;
      yoff = i*ywith +  100;
      gdImageFilledRectangle(im, XDIM+xoff, YDIM-yoff, XDIM+xwith, YDIM-yoff+ywith, color[i]);
      sprintf(str, " %s",rain[i]);
      gdImageStringFT(im, &brect[0], color[22], f, 12.0, 0.0,XDIM+xwith+5 , YDIM-yoff+30, str);
  }
  sprintf(str, "    mm/hr");
```

```
  gdImageStringFT(im, &brect[0], color[22], f, 15.0, 0.0,XDIM+45 ,180,str);

  return 0;
}

 /*********************************************************
 *         AWS지명 그리기                                  *
 *********************************************************/
int aws_draw( gdImagePtr im, int color[])
{
   map.first = 0;
   FILE *fpAws;
   int num_aws;
   char aws_name[50];;
   float aws_lat, aws_lon;
   int i;
   int brect[8];
   char *f = "../font/gulim.ttc";

   fpAws = fopen("../aws/aws_pos_han_2.txt","r");

   fscanf(fpAws,"%d",&num_aws);

   for (i = 0; i < num_aws ; i++)
   {
        fscanf(fpAws,"%s %f %f",aws_name,&aws_lat,&aws_lon);
        alon=aws_lon;
        alat=aws_lat;
        lamcproj(&alon, &alat, &x2, &y2, 0, map);
        if (x2 > 0 && x2 < XDIM && y2 > 0 && y2 < YDIM)
        {
          gdImageStringFT(im, &brect[0], color[22], f, 10.0, 0.0,(int)x2 ,YDIM-(int)y2, aws_name);
        }
   }
  fclose(fpAws);

  return 0;
}
 /*********************************************************
 *         지도 그리기                                     *
 *********************************************************/
int map_draw( gdImagePtr im, int color[])
{
   FILE *fpMap;
   int line, type;
   map.first = 0;
   float aws_lat, aws_lon;
   int i, j;
   float _x=0, _y=0;
   char str[255];

   fpMap = fopen ("../gis/fine_kr.dat" , "r");

   while (fgets(str, 1024, fpMap) != NULL)
```

```
    {
        j = 0;
        sscanf(str, "%d %d", &line, &type);

        fgets(str, 1024, fpMap);
        sscanf(str, "%f %f", &aws_lon, &aws_lat);
        alon=aws_lon;
        alat=aws_lat;
        lamcproj(&alon, &alat, &x2, &y2, 0, map);
        _x = x2;
        _y = y2;

        for (i=0; i<line - 1; i++)
        {
          fgets(str, 1024, fpMap);

          if (type != 3)
          {
            sscanf(str, "%f %f", &aws_lon, &aws_lat);
            alon=aws_lon;
            alat=aws_lat;
            lamcproj(&alon, &alat, &x2, &y2, 0, map);
            if (x2 > 0 && x2 < XDIM && y2 > 0 && y2 < YDIM)
            {
                gdImageLine(im, (int)(x2+0.5), YDIM-(int)(y2+0.5)-1, (int)(_x+0.5),
           YDIM-(int)(_y+0.5)-1, color[22]);
            }
           _x = x2;
           _y = y2;
          }
        }
    }
    fclose(fpMap);
    return 0;
}
//=========================================================//
int date_time( char *filename )
{
  char temp[4];
  int strcnt;
  strcnt = strlen(filename);
  sprintf(temp,"%c%c%c%c",filename[strcnt-19],filename[strcnt-18],filename[strcnt-17],filename[strcnt-16]);
  YY=atoi(temp);
   sprintf(temp,"%c%c", filename[strcnt-15], filename[strcnt-14]);
  MM=atoi(temp);
   sprintf(temp,"%c%c", filename[strcnt-13], filename[strcnt-12]);
    DD=atoi(temp);
        sprintf(temp,"%c%c", filename[strcnt-11], filename[strcnt-10]);
    HH=atoi(temp);
        sprintf(temp,"%c%c", filename[strcnt-9], filename[strcnt-8]);
    MI=atoi(temp);
    printf(" == %04d%02d%02d %02d:%02d == ₩n",YY,MM,DD,HH,MI);
        return 0;
}
```

4.2 Cappi 3D 합성자료

o. Cappi 3D합성자료는 4.1절의 Cappi 합성자료와 같으며 250m 부터 10km까지 250m간격으로 40개 층과 41번째는 관측 영역자료이다.

4.3 480km 합성장

4.3.1 자료 저장

o 레이더 데이터 : 526 * 576 * 4bytes(float) : 강우강도(mm/hr)
- no echo : -999.0

o 관측영역 데이터 : 526 * 576 * 1bytes(unsigned char)
- 관측영역 내 : 1
- 관측영역 외 : 0

4.3.2 지도 좌표 형식

o 지도형식 : Lambert conformal conic projection
o 격자간격 : 3km
o 표준위도 : 30~60
o 기준점위경도 : 126/38
o 기준점XY좌표: 255/360

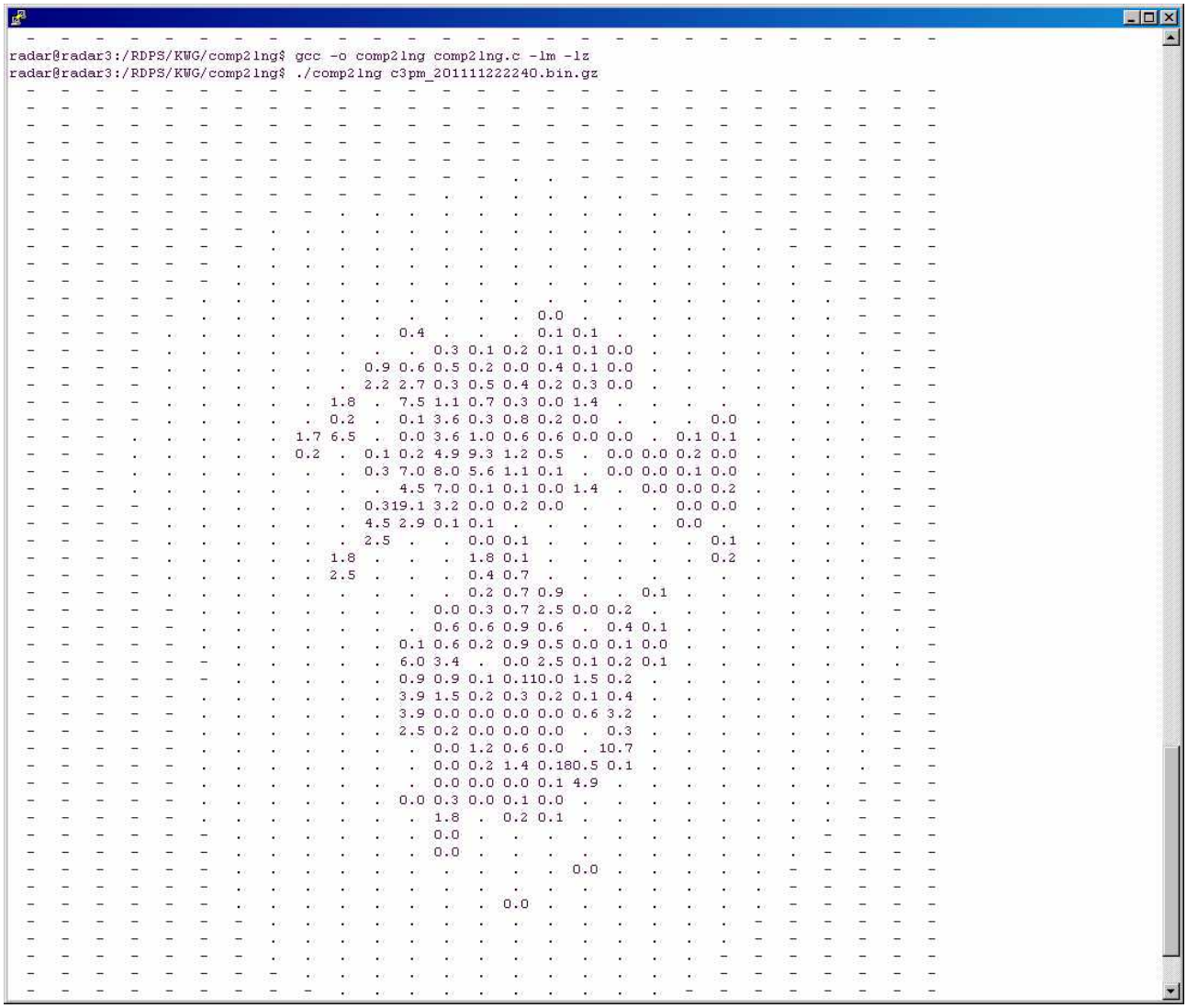
4.3.3 파일 이름 형식

o c3pm_년월일시분.bin.gz(합성 3km, ppi)
※저장순서 : 왼쪽위 -> 오른쪽아래, 가로 우선
※ y값 역순치환 시

4.3.4 자료구조

구분	크기(bytes)	비고
데이터	526 * 576 * 4	float
관측영역	526 * 576 * 1	unsigned char
소계	1,514,880	

```
/************************************************************
**  project : comp2lng.c                                   **
**  comfile : gcc -o comp2lng comp2lng.c -lm -lz           **
************************************************************/
#include <string.h>
#include <stdlib.h>
#include <stdio.h>
#include <math.h>
#include <zlib.h>
#include <zconf.h>

#define XDIM_3 526
#define YDIM_3 576
// XOFF=255
// YOFF=360

// Y... ..
typedef struct
{
  float  data[YDIM_3][XDIM_3]; // unit : mm/hr(noecho = -9999.0)
  unsigned char in_bound[YDIM_3][XDIM_3]; // 관측 범위 데이터(0:out, 1:in)
}COMPOSITION_3KM;

int main(int argc, char *argv[])
{
  FILE  *fp1;
  int i, j, k;
  float dbz, z, rain;
  COMPOSITION_3KM  composition_3km;

  fp1 = gzopen(argv[1],"rb");
  //radar@radar1:/home/racos.noqc/data/composition/data/3km_long/c3pm_201106131340.bin.gz
  gzread(fp1, &composition_3km, sizeof(COMPOSITION_3KM));

        for(j = YDIM_3-1 ; j > 0; j-=10)
        {
                for(i = 0; i < XDIM_3; i+=20)
                {
                        if (composition_3km.data[j][i] != -9999.0) {
                            printf("%4.1f",composition_3km.data[j][i]);
                        }
                        else if (composition_3km.in_bound[j][i] == 1) {
                                printf("   . ");

                        }
                        else {
                                printf("   - ");
                        }
                }

                printf("₩n");
        }

  gzclose(fp1);

}
```


radar@radar3:/RDPS/KWG/comp2lng$ gcc -o comp2lng comp2lng.c -lm -lz
radar@radar3:/RDPS/KWG/comp2lng$ ./comp2lng c3pm_201111222240.bin.gz

4.4 RAR 합성장

4.4.1 자료 저장

o 레이더 데이터 : 1241 * 1761 * 6bytes
- data : 4bytes(float) ,강우강도(mm/hr), noecho는 -9999.0
- 관측영역 : 1bytes(unsigned char)
- 지도 : 1bytes(unsigned char)

4.4.2 지도 좌표 형식

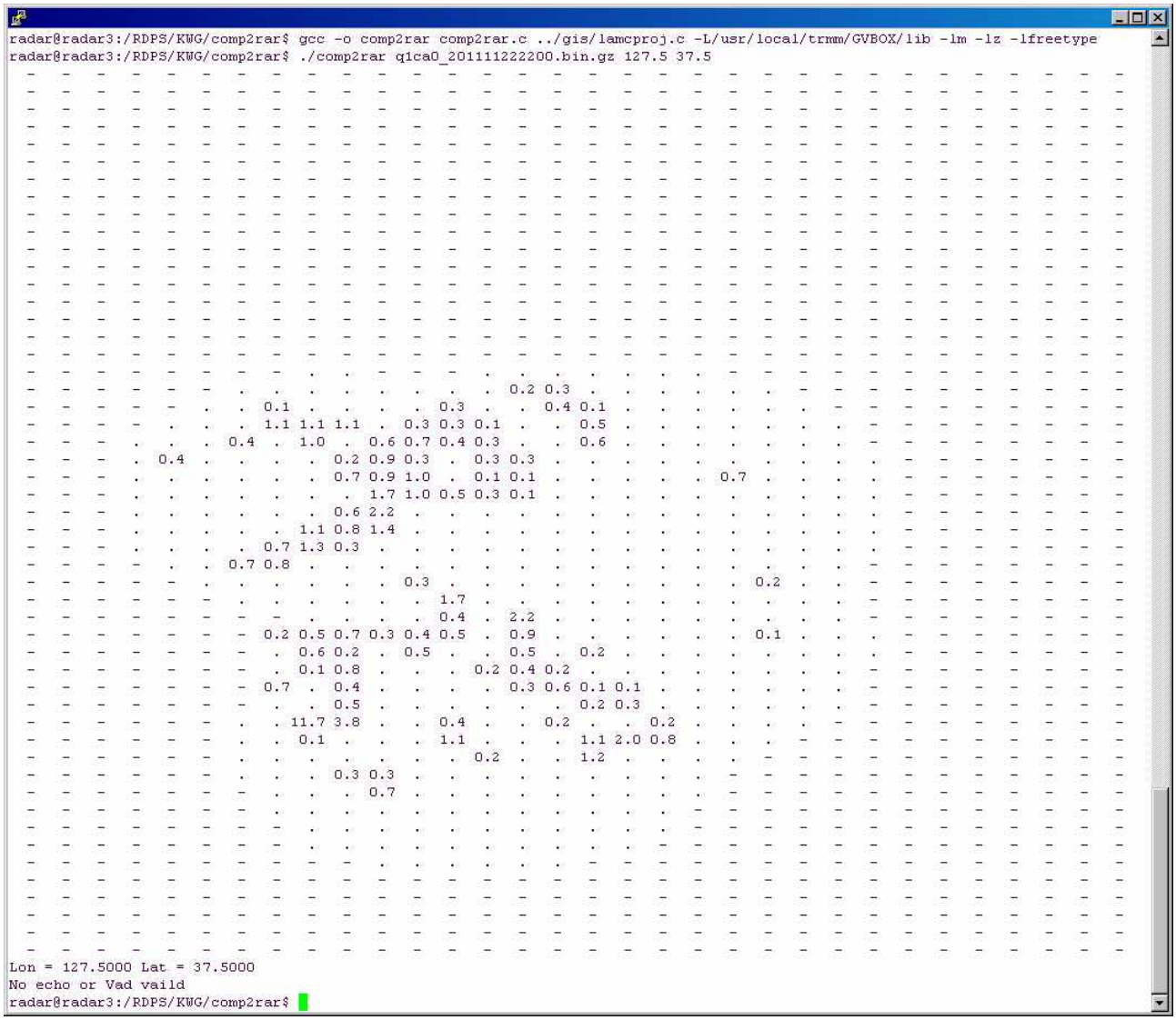
o 지도형식 : Lambert conformal conic projection
o 격자간격 : 3km
o 표준위도 : 30~60
o 기준점위경도 : 126/38
o 기준점XY좌표: 460/925

4.4.3 파일 이름 형식

o q1ca0_년월일시분.bin.gz(합성 1.5km, cappi RAR)
※저장순서 : 왼쪽위 -> 오른쪽아래, 가로 우선
※ y값 역순치환 시

```
/*************************************************************
**  gcc -o read_rar read_rar.c ../gis/lamcproj.c    ₩       **
**  -L/usr/local/trmm/GVBOX/lib -lm -lz -lfreetype          **
*************************************************************/
#include <string.h>
#include <stdlib.h>
#include <stdio.h>
#include <math.h>
#include <zlib.h>
#include "../gis/map_ini.h"
//#include <zconf.h>
#define  XDIM  (1241)
#define  YDIM  (1761)

int main(int argc, char *argv[])
{
  FILE  *fp1;
  int i,j,k;
  float  x2, y2;
  float  alon, alat;
  struct lamc_parameter map;

  float          DATA[YDIM][XDIM];      // noecho = -9999.0
  unsigned char DOUT[YDIM][XDIM];      // rang in =1, out=0
  //unsigned char DMAP[YDIM][XDIM];    //실제 자료가 몽땅 "0"임

  map.Re        = 6370.19584;  // 사용할 지구반경 [ km ]
```

```
  map.grid      = 1.0;          // 격자간격          [ km ]
  map.slat1     = 30.0;         // 표준위도          [degree]
  map.slat2     = 60.0;         // 표준위도          [degree]
  map.olat      = 38.0;         // 기준점의 경도     [degree]
  map.olon      = 126.0;        // 기준점의 위도     [degree]
  map.xo        = 460;          // 기준점의 X좌표    [격자거리]
  map.yo        = 925;          // 기준점의 Y좌표    [격자거리]
  map.first     = 0;            // 시작여부 (0 = 시작)

  fp1 = gzopen(argv[1],"rb");
  gzread(fp1, &DATA, sizeof(DATA));
  gzread(fp1, &DOUT, sizeof(DOUT));
  gzclose(fp1);

  for (j=1;j<YDIM;j+=25)
  {
     for (i=0;i<XDIM;i+=30)
          {
                if(DATA[YDIM-j][i] == -9999.0 && DOUT[YDIM-j][i] == 0)
                {
                    printf("   - ");
                }
                else if(DATA[YDIM-j][i] == -9999.0 && DOUT[YDIM-j][i] == 1)
                {
                    printf("   . ");
                }
                else
                {
                    printf("%4.1f",DATA[YDIM-j][i]);
                }
     }
     printf("₩n");
  }

  alon=atof(argv[2]); //aws_lon;
  alat=atof(argv[3]); //aws_lat;
  lamcproj(&alon, &alat, &x2, &y2, 0, map);

  printf("Lon = %7.4f Lat = %7.4f ₩n",alon, alat);

  if (DATA[(int)y2][(int)x2] != -9999.0 && DOUT[(int)y2][(int)x2] == 1)
  {
        printf("[%03d][%03d] = %5.1f mm/hr ₩n",(int)y2,(int)x2, DATA[(int)y2][(int)x2]);
  }
  else
  {
        printf("No echo or Vad vaild ₩n");
  }

}
```


radar@radar3:/RDPS/KWG/comp2rar$ gcc -o comp2rar comp2rar.c ../gis/lamcproj.c -L/usr/local/trmm/GVBOX/lib -lm -lz -lfreetype
radar@radar3:/RDPS/KWG/comp2rar$ ./comp2rar q1ca0_201111222200.bin.gz 127.5 37.5
Lon = 127.5000 Lat = 37.5000
No echo or Vad vaild
radar@radar3:/RDPS/KWG/comp2rar$

4.5 WPMM 합성장

4.5.1 자료 저장

o 레이더 데이터 : 901 * 1051 * 2bytes(short) : 강우강도(mm/hr)
- echo : value/100 (mm/hr)
- no echo : -32767
- 관측영역 외곽 : -32768

4.5.2 지도 좌표 형식

o 지도형식 : Lambert conformal conic projection
o 격자간격 : 3km
o 표준위도 : 30~60
o 기준점위경도 : 126/38
o 기준점XY좌표: 366/771

4.5.3 파일 이름 형식

o RDR_CMP_년월일시분.bin.gz(합성 1.5km, cappi)
※저장순서 : 왼쪽위 -> 오른쪽아래, 가로 우선
※ y값 역순치환 시

```
/*****************************************************************
**  gcc -o comp2wpmm comp2wpmm.c ../gis/lamcproj.c   ₩     **
**  -L/usr/local/trmm/GVBOX/lib -lrsl -lm -lz -lfreetype        **
*****************************************************************/
#include <string.h>
#include <stdlib.h>
#include <stdio.h>
#include <math.h>
#include <zlib.h>
#include "../gis/map_ini.h"
#define  XDIM  (901)
#define  YDIM  (1051)

static void swap2(void *word);
static void swap4(void *word);

int main(int argc, char *argv[])
{
  FILE  *fp1;
  int i,j,k;
  float  x2, y2;
  float  alon, alat;
  struct lamc_parameter map;

  short DATA[YDIM][XDIM];    // noecho = -32767

  map.Re        = 6370.19584;  // ... .... [ km ]
```

```
  map.grid        = 1.0;            // ....          [ km ]
  map.slat1       = 30.0;           // ....          [degree]
  map.slat2       = 60.0;           // ....          [degree]
  map.olat        = 38.0;           // .... ..    [degree]
  map.olon        = 126.0;          // .... ..    [degree]
  map.xo          = 366;            // .... X..   [....]
  map.yo          = 771;            // .... Y..   [....]
  map.first       = 0;              // .... (0 = ..)

  fp1 = gzopen(argv[1],"rb");
  gzread(fp1, &DATA, sizeof(DATA));
  gzclose(fp1);

  for (j=1;j<YDIM;j+=35)
  {
        for (i=0;i<XDIM;i+=40)
        {
              swap2(&DATA[YDIM-j][i]);
              if(DATA[YDIM-j][i] == -32768)
              {
                    printf("   - ");
              }
              else if(DATA[YDIM-j][i] == -32767)
              {
                    printf("   . ");
              }
              else
              {
                    printf("%4.1f",(float)DATA[YDIM-j][i]/10);
              }
        }
        printf("₩n");
  }

  alon=atof(argv[2]); //aws_lon;
  alat=atof(argv[3]); //aws_lat;
  lamcproj(&alon, &alat, &x2, &y2, 0, map);

  printf("Lon = %7.4f Lat = %7.4f ₩n",alon, alat);

  if (DATA[(int)y2][(int)x2] > 0 )
  {
       printf("[%03d][%03d] = %5.1f mm/hr ₩n",(int)y2,(int)x2, DATA[(int)y2][(int)x2]/10);
  }
  else
  {
       printf("No echo or Vad vaild ₩n");
  }

}

static void swap2(void *word)
{
  unsigned char *byte;
  unsigned char temp;
  byte     = (unsigned char *)word;
  temp     = byte[0];
  byte[0] = byte[1];
  byte[1] = temp;
}
static void swap4(void *word)
{
  unsigned char *byte;
  unsigned char temp;
```

```
  byte     = (unsigned char *)word;
  temp     = byte[0];
  byte[0] = byte[3];
  byte[3] = temp;
  temp     = byte[1];
  byte[1] = byte[2];
  byte[2] = temp;
}
```

```
  -   -   -   -   -   .  2.7   .  3.5 0.7 6.2   .  9.2 0.7   .   .   .   .   .   .   .   .   -
  -   -   -   -   -   . 10.3 5.6   .   .   .  1.5 6.6 3.1   .   .   .   .   .   .   .   .   -
  -   -   -   -   .  7.6 0.8 1.8   .   .   .  5.7 8.911.3 1.7   .   .   .   .   .   .   .   -
  -   -   -   -   .   . 86.029.1   .   .   .   .   .   .  2.7 3.0   .   .   .   .   .   .   -
  -   -   -   -   .   . 106.8 5.6   .   .   .   .   .  2.2   .  1.4 1.8   .   .   .   .   -   -
  -   -   -   -   .   .   .   .   .   .  6.1   .   .   . 12.318.915.3   .   .   .   -   -   -
  -   -   -   -   .  3.0   .   .   .  3.145.5 5.0 2.9   .  3.8   .   .   .   .   -   -   -   -
  -   -   -   -   .   .   .   .  4.7 1.4   .   .   .   .   .   .   .   .   .   -   -   -   -   -
  -   -   -   -   .   .   . 99.9 2.3   .   .   .   .   .   .   .   .   .   -   -   -   -   -   -
  -   -   -   -   -   .   .   .   .   .   .   .   .   .   .   .   .   .   -   -   -   -   -   -
  -   -   -   -   -   .   .  118.9   .   .   .   .   .   .   .   .   .   -   -   -   -   -   -   -
  -   -   -   -   -   -   .   .   .   .   .   .   .   .   .   .   -   -   -   -   -   -   -   -
  -   -   -   -   -   -   -   .   .   .   .   .   .   .   .   -   -   -   -   -   -   -   -   -
Lon = 127.5000 Lat = 37.5000
[718][495] = 127.5 mm/hr
radar@radar3:/RDPS/KWG/comp2wpmm$ gcc -o comp2wpmm comp2wpmm.c ../gis/lamcproj.c -L/usr/local/trmm/GVBOX/lib -lrsl -lm -lz -lfreet
ype
radar@radar3:/RDPS/KWG/comp2wpmm$ ./comp2wpmm RDR_CMP_201111222210.bin.gz 127.5 37.5
  -   -   -   -   -   -   -   -   -   -   -   -   -   -   -   -   -   -   -   -   -   -   -   -   -
  -   -   -   -   -   -   -   -   -   -   -   -   -   -   -   -   -   -   -   -   -   -   -   -   -
  -   -   -   -   .   .   .   .   .   .   .   .   .   .   .  1.3   .   .   .   .   .   .   -   -   -
  -   -   -   .   .   .   .  0.3   .   .   .  4.6 2.0 0.9 1.9 2.1   .   .   .   .   .   .   -   -   -
  -   -   .   .   .   .   .   .  6.4   .   .  1.8   .  2.4   .  5.1   .   .   .   .   .   .   .   -   -
  -   -   .   .   .  7.4   .   .  7.5 2.1 5.3 6.9 4.2 1.8   .  4.6 1.4   .   .   .   .   .   .   -   -
  -   .   .   .   .   .   . 30.1 5.0 4.415.6 3.0   .   .   .   .  5.7   .   .   .  5.4   .   .   .   .   -
  -   .   .   .   .   .   .   .  5.1 9.8 4.3   .  1.6 2.9   .   .   .   .   .   .   .   .   .   .   -
  -   .   .   .   .   .   .   .  7.229.510.4   .  2.7 6.0   .   .   .   .   .  4.6   .   .   .   .   .
  .   .   .   .   .   .   .   .  1.132.711.3 2.3 2.1 2.3   .   .   .   .   .   .   .   .  0.5   .   .   .
  .   .   .   .   .   .   . 10.7 9.854.315.1 1.5 6.3   .   .   .   .   .   .   .   . 2.7 0.9   .   .   .
  -   .   .   .   .   .   . 15.119.721.7 0.4   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .
  -   .   .   .   .   .  . 66.548.545.0 8.5   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .
  -   .   .   .   .  3.115.618.9 0.9   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   -
  -   -   .   .  3.8 3.5   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   -
  -   -   .   . 10.6   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .  9.6   .   -   -
  -   -   -   .   .   .   .   .   .   .   .  3.7   .   .   .   .   .   .   .   .   .   .   .   .   -
  -   -   -   -   -   .   .   .   .   .   .  2.3 5.7 2.1   .   .   .   .   .   .   .   .   .   .   -
  -   -   -   -   -   -   .   .   .   .   .  2.0   . 10.012.5   .   .   .   .   .   .   .   .   -
  -   -   -   -   -   -   .  4.9 6.0 9.1   .   .  2.219.5 6.4   .   .   .   .   .   .   .   .   -
  -   -   -   -   -  0.6 1.2 1.3 4.1   . 12.3   .  5.211.612.5   .   .   .   .   .   .   .   .   -
  -   -   -   -   -   .  8.715.3 5.6   .   .   .   .  8.9 4.1   .   .   .   .   .   .   .   .   -
  -   -   -   -   -   .  2.712.5 1.3   .   .   .   .  4.913.822.1 1.2   .   .   .   .   .   .   -
  -   -   -   -   -   .   .  5.8 9.3   .   .   .   .  2.1 9.5   .  1.812.6 0.5   .   .   .   .   .   -
  -   -   -   -   -   .   . 44.6   .   .   .   .   .   .   .   .  1.1   .   .   .   .   .   .   -   -
  -   -   -   -   .   .   .  2.224.9 8.9   .   .   .   .   .   .   .  7.110.9   .   .   .   .   .   -   -
  -   -   -   -   .   .   .   .   .   .   . 12.0   .   .   .   . 30.331.913.0   .   .   .   .   -   -   -
  -   -   -   -   .   .   .   .   .   .  7.8   .  3.1   .   . 46.6 1.7   .   .   .   .   -   -   -   -
  -   -   -   -   .   .   .   .   .  4.7 3.7   .   .   .   .  1.6   .   .   .   .   .   -   -   -   -   -
  -   -   -   -   -   .   .   . 30.5   .   .   .   .   .   .   .   .   .   .   .   -   -   -   -   -   -
  -   -   -   -   -   .   .   . 609.1   .   . 16.2   .   .   .   .   .   .   .   -   -   -   -   -   -   -
  -   -   -   -   -   .   .   .   .   .   .   .   .   .   .   .   .   .   .   -   -   -   -   -   -   -
  -   -   -   -   -   -   .   .   .   .   .   .   .   .   .   .   .   .   .   -   -   -   -   -   -   -
  -   -   -   -   -   -   -   .   .   .   .   .   .   .   .   .   .   .   .   -   -   -   -   -   -   -
  -   -   -   -   -   -   -   -   -   .   .   .   .   .   .   .   -   -   -   -   -   -   -   -   -   -
Lon = 127.5000 Lat = 37.5000
[718][495] = 127.5 mm/hr
radar@radar3:/RDPS/KWG/comp2wpmm$
```

제5장 maple자료

5.1 자료 저장

o 레이더 데이터 : 1024 * 1024 * 37 : 4bytes(float) : 강우강도 (mm/hr)

- 관측반경 밖 : -10.0
- no echo : 0

5.2 지도 좌표 형식

o 지도형식 : Lambert conformal conic projection
o 격자간격 : 1km
o 표준위도 : 30~60
o 기준점위경도 : 126/38
o 기준점XY좌표: 427/771

5.3 파일 이름 형식

o Z=1(10분 후 예측장)
o Z=2(20분 후 예측장)
…
o Z=36(360분 후 예측장)
o Z=37(초기장)

※저장순서 : 왼쪽위 -> 오른쪽아래, 가로 우선
※ y값 역순치환 시

5.4 자료구조

구분	크기(bytes)	비고
Forecast_hed	44	
Arrayhed	60	
data	155189248	1024*1024*37*4bytes
Compvel_hed	36	
uvbuf	5000	
소계	155,194,388	

5.5 자료변환 예시

```
/************************************************************
** gcc -o read_maple read_maple.c ../gis/lamcproj.c -lm -lz    **
** info : kim.wongi  019-9173-0365                            **
************************************************************/
#include<sys/types.h>
#include<sys/stat.h>
#include<fcntl.h>
#include<errno.h>
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include<zlib.h>
#include "maple.h"
#include "../gis/map_ini.h"

#define XDIM 1024 //X축 격자수 km
#define YDIM 1024 //Y축 격자수 km
#define ZDIM 37   //예측장 수

#define VROWS 25
#define VCOLS 25

float PDATA[XDIM][YDIM][ZDIM];
float UVA[VROWS][VCOLS][2];

struct lamc_parameter map;
float alon, alat, x2, y2;

int main(int argc, char *argv[])
{
 int i,j, irow, icol;
 int poin;
 FILE *fp;
 char fname[200];
 long icount;

  map.Re        = 6370.19584;  // 사용할 지구반경 [ km ]
  map.grid      = 1.0;      // 격자간격         [ km ]
  map.slat1     = 30.0;     // 표준위도         [degree]
  map.slat2     = 60.0;     // 표준위도         [degree]
  map.olat      = 38.0;     // 기준점의 경도    [degree]
  map.olon      = 126.0;    // 기준점의 위도    [degree]
  map.xo        = 427;      // 기준점의 X좌표   [격자거리]
  map.yo        = 771;      // 기준점의 Y좌표   [격자거리]
  map.first     = 0;        // 시작여부 (0 = 시작)

 Forecast_hed forecast_hed;
 Arrayhed              ahed;
 Compvel_hed    compvel_hed;

 printf("file name : %s₩n",argv[1]);
```

```
 fp = gzopen(argv[1],"rb"); //압축화일 읽기

 gzread(fp,&forecast_hed,sizeof(Forecast_hed));
        for (i=0; i<10; i++)
        {
          printf("forecast_period[%i]=%i₩n",i,forecast_hed.forecast_period[i]);
        }
        printf("nforecasts =%i₩n",forecast_hed.nforecasts);
  if (forecast_hed.nforecasts == 0)
  {
          gzclose(fp);
          exit(0);
  }

 gzread(fp,&ahed,sizeof(Arrayhed));
        printf("=======================================₩n");
        printf("Print the header for rainrate info.₩n");
        printf("Year     : %i₩n",ahed.year);
        printf("Month    : %i₩n",ahed.month);
        printf("Day      : %i₩n",ahed.day);
        printf("Hours    : %i₩n",ahed.hours);
        printf("Minutes  : %i₩n",ahed.minutes);
        printf("nx       : %i₩n",ahed.nx);
        printf("ny       : %i₩n",ahed.ny);

 gzread(fp,PDATA,sizeof(PDATA));

/*
 for (i=0; i<ZDIM; i++)
 {
    for (irow = 0; irow < XDIM; irow++)
    {
        for (icol = 0; icol < YDIM; icol++)
        {
            if (PDATA[irow][icol][i] > 0)
            {
                printf("PDATA[%i][%i][%i] = %5.1f₩n",irow,icol,i,PDATA[irow][icol][i]);
                x2=(float)irow;
                y2=(float)icol;
                lamcproj(&alon, &alat, &x2, &y2, 1, map);
                printf(" lon=%7.4f lat=%7.4f ₩n",alon,alat);
            }
        }
    }
  }

*/

 gzread(fp,&compvel_hed, sizeof(Compvel_hed));

        printf("=======================================₩n");
        printf("Print the header for velocity info.₩n");
        printf("Year     : %i₩n",compvel_hed.year);
        printf("Month    : %i₩n",compvel_hed.month);
```

```
        printf("Day       : %i₩n",compvel_hed.day);
        printf("Hours     : %i₩n",compvel_hed.hours);
        printf("Minutes : %i₩n",compvel_hed.minutes);
        printf("Seconds : %i₩n",compvel_hed.seconds);
        printf("velf_ew_size : %i₩n",compvel_hed.velf_ew_size);
        printf("velf_sn_size : %i₩n",compvel_hed.velf_sn_size);

 gzread(fp,UVA,sizeof(UVA));
  /*
  for(i=0; i <  11; i++)
  {
       for (irow=0; irow <  VROWS; irow++)
       {
            for(icol=0; icol <  VCOLS; icol++)
            {
                 printf("UVA[%i][%i][%d] : %f₩n",irow,icol,i, UVA[irow][icol][i]);
            }
       }
   }
  */
 gzclose(fp);
 return 0;
}
```

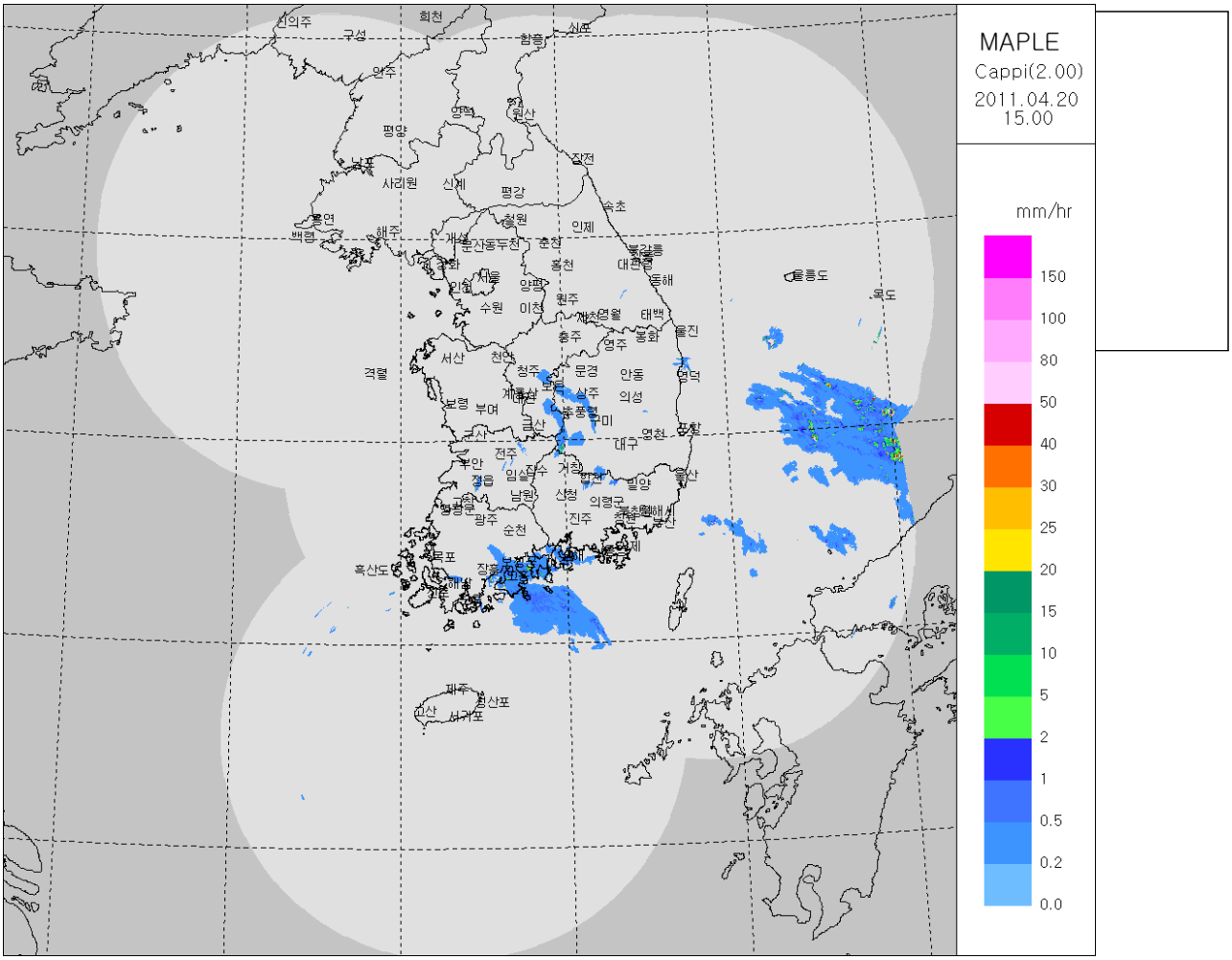
```
@radar3:/RDPS/KWG/maple$ ./read_maple RDR_MAPLE_CAPPI20_RAIN_201009211200.bin.gz|more
file name : RDR_MAPLE_CAPPI20_RAIN_201009211200.bin.gz
forecast_period[0]=600
forecast_period[1]=1024
forecast_period[2]=1024
forecast_period[3]=1000
forecast_period[4]=0
forecast_period[5]=0
forecast_period[6]=0
forecast_period[7]=0
forecast_period[8]=0
forecast_period[9]=0
nforecasts =36
=======================================
Print the header for rainrate info.
Year     : 2010
Month    : 9
Day      : 21
Hours    : 12
Minutes : 0
nx       : 1024
ny       : 1024
=======================================
Print the header for velocity info.
Year     : 2010
Month    : 9
```

```
Day      : 21
Hours    : 12
Minutes : 0
Seconds : 0
velf_ew_size : 25
velf_sn_size : 25
```

```
/*** maple.h **/
/*    structure for time  */
typedef  struct {
  int forecast_period[10],nforecasts;
}Forecast_hed;
typedef struct {
  int  year,month,day,hours,minutes;    /*** time of composite ***/
  int  nx,ny;    /*** dimensions in west-east and north-south directions ***/
  float  clat,clon,yminl,ymaxl,xminl,xmaxl;
  int forecast_period,maptype;  /*** these two parameters are required for
  McGill display ***/
} Arrayhed;
typedef  struct {
  short int seconds;
  short int minutes;
  short int hours;
  short int day;
  short int month;
  short int year;
  float pixel_res_km;
  int  border_pix,ew_arsize,sn_arsize,velf_ew_size,velf_sn_size;
}Compvel_hed;
```

o Maple자료 영상 생성

```
/*****************************************************************
 **  gcc -o maple_3d maple_3d.c function.c ../gis/lamcproj.c  ₩  **
 **  -lm -lgd -lpng -lz                                          **
 *****************************************************************/
#include <string.h>
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[])
{
   int i, j, k;
```



[maple 자료 영상]

```
//filename : function.c

#include <string.h>
#include <stdio.h>
#include <stdlib.h>
#include "maple.h"
//#include "../gis/map_ini.h"

int maple_to_png(char *filename, int dap)
{
    gdImagePtr im;
    FILE *pngout;

    struct lamc_parameter map;
    float alon, alat, x2, y2;

    int color[255];
    int i, j, k;
    int YY, MM, DD, HH, MI;

    map.Re    = 6370.19584;  // 사용할 지구반경 [ km ]
    map.grid  = 1.0;         // 격자간격        [ km ]
    map.slat1 = 30.0;        // 표준위도        [degree]
    map.slat2 = 60.0;        // 표준위도        [degree]
```

```
   map.olat   = 38.0;          // 기준점의 경도    [degree]
   map.olon   = 126.0;         // 기준점의 위도    [degree]
   map.xo     = 427;           // 기준점의 X좌표   [격자거리]
   map.yo     = 771;           // 기준점의 Y좌표   [격자거리]
   map.first  = 0;             // 시작여부 (0 = 시작)

   im = gdImageCreate(XDIM+150, YDIM);  //이미지 생성

     color[0]   = gdImageColorAllocate(im, 110, 190, 255);
     color[1]   = gdImageColorAllocate(im,  62, 148, 255);
     color[2]   = gdImageColorAllocate(im,  62, 116, 255);
     color[3]   = gdImageColorAllocate(im,  40,  49, 255);
     color[4]   = gdImageColorAllocate(im,  70, 255,  70);
     color[5]   = gdImageColorAllocate(im,   0, 224,  80);
     color[6]   = gdImageColorAllocate(im,   0, 175, 101);
     color[7]   = gdImageColorAllocate(im,   0, 150, 101);
     color[8]   = gdImageColorAllocate(im, 255, 229,   0);
     color[9]   = gdImageColorAllocate(im, 255, 190,   0);
     color[10]  = gdImageColorAllocate(im, 255, 112,   0);
     color[11]  = gdImageColorAllocate(im, 215,   0,   0);
     color[12]  = gdImageColorAllocate(im, 255, 206, 255);
     color[13]  = gdImageColorAllocate(im, 255, 170, 255);
     color[14]  = gdImageColorAllocate(im, 255, 125, 250);
     color[15]  = gdImageColorAllocate(im, 255,   0, 255);
     color[16]  = gdImageColorAllocate(im, 136, 136, 136);
     color[17]  = gdImageColorAllocate(im, 163, 163, 163);
     color[18]  = gdImageColorAllocate(im, 110, 190, 255);
     color[230] = gdImageColorAllocate(im, 247, 150,  70);
     color[231] = gdImageColorAllocate(im, 202, 107, 130);
     color[240] = gdImageColorAllocate(im,   0,   0,   0);
     color[250] = gdImageColorAllocate(im, 224, 224, 224);
     color[251] = gdImageColorAllocate(im, 255, 255, 255);
     color[252] = gdImageColorAllocate(im, 196, 196, 196);
     color[253] = gdImageColorAllocate(im,   0,   0,   0);
     color[254] = gdImageColorAllocate(im, 149, 149, 149);
     color[255] = gdImageColorAllocate(im, 196, 196, 196);

   gdImageFilledRectangle(im, 0, 0, XDIM, YDIM, color[250]);  //echo box 배경색


//read_radar=========================================================================
   FILE    *fpRdr;
   int n, cindex;
   float  buf[1024][1024];

   Arrayhed ahed_skill;
   Forecast_hed forecast_hed_skill;
   Compvel_hed compvel_hed;

   fpRdr = gzopen(filename,"rb");
   gzread(fpRdr,&forecast_hed_skill,sizeof(Forecast_hed));
      YY = ahed_skill.year;
      MM = ahed_skill.month;
      DD = ahed_skill.day;
      HH = ahed_skill.hours;
      MI = ahed_skill.minutes;
          //printf(" %04d %02d %02d %02d %02d₩n",YY,MM,DD,HH,MI);
   gzread(fpRdr,&ahed_skill,sizeof(Arrayhed));
   if (dap != 0) gzseek(fpRdr,dap*sizeof(buf), SEEK_CUR);
```

```
gzread(fpRdr, &buf, sizeof(buf));

for (j=0;j<1024;j++)
{
  for (i=1;i<1024;i++)
  {
      switch((int)(buf[j][i]*10.0))
      {
         case -100:
             cindex = 252; //관측외곽
             break;
         case 1 ... 2:
             cindex = 1;
             break;
         case 3 ... 5:
             cindex = 2;
             break;
         case 6 ... 10:
             cindex = 3;
             break;
         case 11 ... 20:
             cindex = 4;
             break;
         case 21 ... 50:
             cindex = 5;
             break;
         case 51 ... 100:
             cindex = 6;
             break;
         case 101 ... 150:
             cindex = 7;
             break;
         case 151 ... 200:
             cindex = 8;
             break;
         case 201 ... 250:
             cindex = 9;
             break;
         case 251 ... 300:
             cindex = 10;
             break;
         case 301 ... 400:
             cindex = 11;
             break;
         case 401 ... 500:
             cindex = 12;
             break;
         case 501 ... 800:
             cindex = 13;
             break;
         case 801 ... 1000:
             cindex = 14;
             break;
         case 1001 ... 1500:
             cindex = 15;
             break;
         default:
             cindex = 250;
      }
```

```
        gdImageSetPixel(im, i, j, color[cindex]);
    }
}
gzclose(fpRdr);

// latlon_draw ======================
int   lon, lat;
float   x1, y1;

// 경도선 /
for (lon = 120; lon < 140; lon += 2)
{
   alon = (float)lon;
   alat = 80;
   lamcproj(&alon, &alat, &x1, &y1, 0, map);

   alon = (float)lon;
   alat = 0;
   lamcproj(&alon, &alat, &x2, &y2, 0, map);
   gdImageDashedLine(im, (int)x1, YDIM-(int)y1, (int)x2, YDIM-(int)y2, color[240]);
}

// 위도선 /
for (lat = 30; lat < 50; lat += 2)
{
   for (lon = 120*10; lon < 140*10; lon += 1)
   {
     alon = (float)lon*0.1;
     alat = (float)lat;
     lamcproj(&alon, &alat, &x1, &y1, 0, map);
     alon = (float)lon*0.1 +  1*0.05;
     alat = (float)lat;
     lamcproj(&alon, &alat, &x2, &y2, 0, map);
     gdImageLine(im, (int)x1, YDIM-(int)y1, (int)x2, YDIM-(int)y2, color[240]);
   }
}


// aws_draw=============================
FILE *fpAws;
char *f = "../font/gulim.ttc";
char aws_name[50];;
int brect[8];
int num_aws;
float aws_lat, aws_lon;
map.first = 0;

fpAws = fopen("../aws/aws_pos_han_2.txt","r");
fscanf(fpAws,"%d",&num_aws);

for (i = 0; i < num_aws ; i++)
{
    fscanf(fpAws,"%s %f %f",aws_name,&aws_lat,&aws_lon);
    alon=aws_lon;
    alat=aws_lat;
    lamcproj(&alon, &alat, &x2, &y2, 0, map);
    if (x2 > 0 && x2 < XDIM && y2 > 0 && y2 < YDIM)
    {
      gdImageStringFT(im, &brect[0], color[253], f, 10.0, 0.0,
```

```
                                        (int)x2,YDIM-(int)y2, aws_name);
        }
    }
    fclose(fpAws);

    // map_draw=========================
    FILE *fpMap;
    char str[255];
    int line, type;
    //float aws_lat, aws_lon;
    float _x=0, _y=0;

    map.first = 0;

    fpMap = fopen ("../gis/fine_kr.dat" , "r");

    while (fgets(str, 1024, fpMap) != NULL)
    {
        j = 0;
        sscanf(str, "%d %d", &line, &type);

        fgets(str, 1024, fpMap);
        sscanf(str, "%f %f", &aws_lon, &aws_lat);
        alon=aws_lon;
        alat=aws_lat;
        lamcproj(&alon, &alat, &x2, &y2, 0, map);
        _x = x2;
        _y = y2;

        for (i=0; i<line - 1; i++)
        {
            fgets(str, 1024, fpMap);
            if (type != 3)
            {
                sscanf(str, "%f %f", &aws_lon, &aws_lat);
                alon=aws_lon;
                alat=aws_lat;
                lamcproj(&alon, &alat, &x2, &y2, 0, map);
                if (x2 > 0 && x2 < XDIM && y2 > 0 && y2 < YDIM)
                {
                    gdImageLine(im, (int)(x2+0.5), YDIM-(int)(y2+0.5)-1,
                    (int)(_x+0.5), YDIM-(int)(_y+0.5)-1, color[253]);
                }
                _x = x2;
                _y = y2;
            }
        }
    }

    fclose(fpMap);


    // title_draw=========================
    //char str[255];
    //char *f = "./include/gulim.ttc";
    char *err;
    //int brect[8];

    gdImageFilledRectangle(im, XDIM, 0, XDIM+150, YDIM, color[251]);  //title box 배경색
```

```
    sprintf(str, "%s","MAPLE");
    err = gdImageStringTTF(im,&brect[0],color[253], f, 20.0,0.0,XDIM+25 , 50, str);

    sprintf(str, "after %02d:%02d", ((dap+1)*10)/60,((dap+1)*10)%60);
    gdImageStringFT(im, &brect[0], color[253], f, 16.0, 0.0,XDIM+20 , 80, str);

    sprintf(str, "%04d.%02d.%02d", YY,MM,DD);
    gdImageStringFT(im, &brect[0], color[253], f, 16.0, 0.0,XDIM+20 , 110, str);
    sprintf(str, "%02d.%02d", HH, MI);
    gdImageStringFT(im, &brect[0], color[253], f, 16.0, 0.0,XDIM+50 , 130, str);

    static char  rain[][15] ={
           "0.0","0.2","0.5","1","2","5","10","15","20","25","30","40","50","80","100","150"};
    int xoff, xwith; //x축 여벽, 폭
    int yoff, ywith; //y축 여벽, 폭
    for (i=0;i<16 ;i++ )
    {
       xoff = 30;
       xwith = 80; //80-30=50
       ywith = 45; //40;
       yoff = i*ywith + 100;
       gdImageFilledRectangle(im, XDIM+xoff, YDIM-yoff, XDIM+xwith, YDIM-yoff+ywith, color[i]);
       sprintf(str, " %s",rain[i]);
       gdImageStringFT(im, &brect[0], color[240], f, 12.0, 0.0,XDIM+xwith+5 , YDIM-yoff+50, str);
    }
    sprintf(str, "    mm/hr");
    gdImageStringFT(im, &brect[0], color[240], f, 15.0, 0.0,XDIM+45 ,230,str);


    gdImageRectangle(im,    0,    0,XDIM,       YDIM-1, color[240]);  //echo box 테두리선
    gdImageRectangle(im, XDIM,    0,XDIM+150-1,YDIM-1, color[240]); //title box 테두리선
    gdImageRectangle(im, XDIM, 150,XDIM+150-1,YDIM-1, color[240]);  //title box 중간선

    //============================================================
    char outfile[100]="";
    char imsi[100]="";
    char pngfile[100]="";

    strncpy(imsi, filename, strlen(filename)-strlen(strchr(filename, '.')));
    sprintf(outfile,"%s.png",imsi);
    sprintf(pngfile, "%02d_%s", dap+1, outfile);
    printf("pngfile is  : %s₩n",pngfile);

    pngout  = fopen(pngfile,"wb");
    gdImagePng(im,pngout);
    fclose(pngout);
    gdImageDestroy(im);
}
```

```
//filename : maple.h
#include <gd.h>
#include <math.h>
```

```
#include <gdfontt.h>
#include <gdfonts.h>
#include <gdfontmb.h>
#include <gdfontl.h>
#include <gdfontg.h>
#include <zlib.h>
#include <zconf.h>
#include <malloc.h>
#include "../include/cgiutil.h"
#include "../include/cgic.h"
#include "../gis/map_ini.h"

#define  XDIM  1024
#define  YDIM  1024

/*** structure for the composite rradar map velocity field  **/
/*    structure for time  */
typedef  struct {
  int forecast_period[10],nforecasts;
}Forecast_hed;
typedef struct {
  int  year,month,day,hours,minutes;    /*** time of composite ***/
  int  nx,ny;    /*** dimensions in west-east and north-south directions ***/
  float  clat,clon,yminl,ymaxl,xminl,xmaxl;
  int forecast_period,maptype;  /*** these two parameters are required for
  McGill display ***/
} Arrayhed;
typedef  struct {
  short int seconds;
  short int minutes;
  short int hours;
  short int day;
  short int month;
  short int year;
  float pixel_res_km;
  int  border_pix,ew_arsize,sn_arsize,velf_ew_size,velf_sn_size;
}Compvel_hed;

int maple_to_png(char *filename, int dap);
```

o. openmpi 병렬처리

```
/*************************************************************************
* comfile : mpicc -I./include maple_3d_mpi.c ./lamcproj.c ./function.c -o ₩ *
*   maple_3d_mpi.x -lgd -lpng -lz                                        *
* run : mpirun -np 37 -hostfile ./host.list ./maple_3d_mpi.x          ₩  *
* RDR_MAPLE_CAPPI20_RAIN_201009211200.bin.gz                             *
*************************************************************************/

#include <stdio.h>
#include "maple.h"
#include <mpi.h>
#define M 50000

int main(int argc, char* argv[])
{
    int i, nprocs, myrank;
    int isend[M], irecv[3];

    MPI_Init(&argc, &argv);
    MPI_Comm_size(MPI_COMM_WORLD, &nprocs);
    MPI_Comm_rank(MPI_COMM_WORLD, &myrank);

    printf(" nprocs = %d myrank = %d ₩n",nprocs,myrank);

    char infile[100];
    sprintf(infile,"%s",argv[1]);

    for (i=0;i<nprocs;i++)
    {
        if( myrank == i)
        {
            maple_to_png(infile, myrank);
            //printf(" %s  %d ₩n",infile, myrank);
        }
    }

    MPI_Finalize();

}
```

제6장 레이더+위성+낙뢰 합성영상 생성

6.1 입력자료

o 레이더 : 960 * 1200 * 1bytes(1km격자 Cappi 합성장)

- Lambert conformal conic projection지도 투영에 매핑된 격자자료

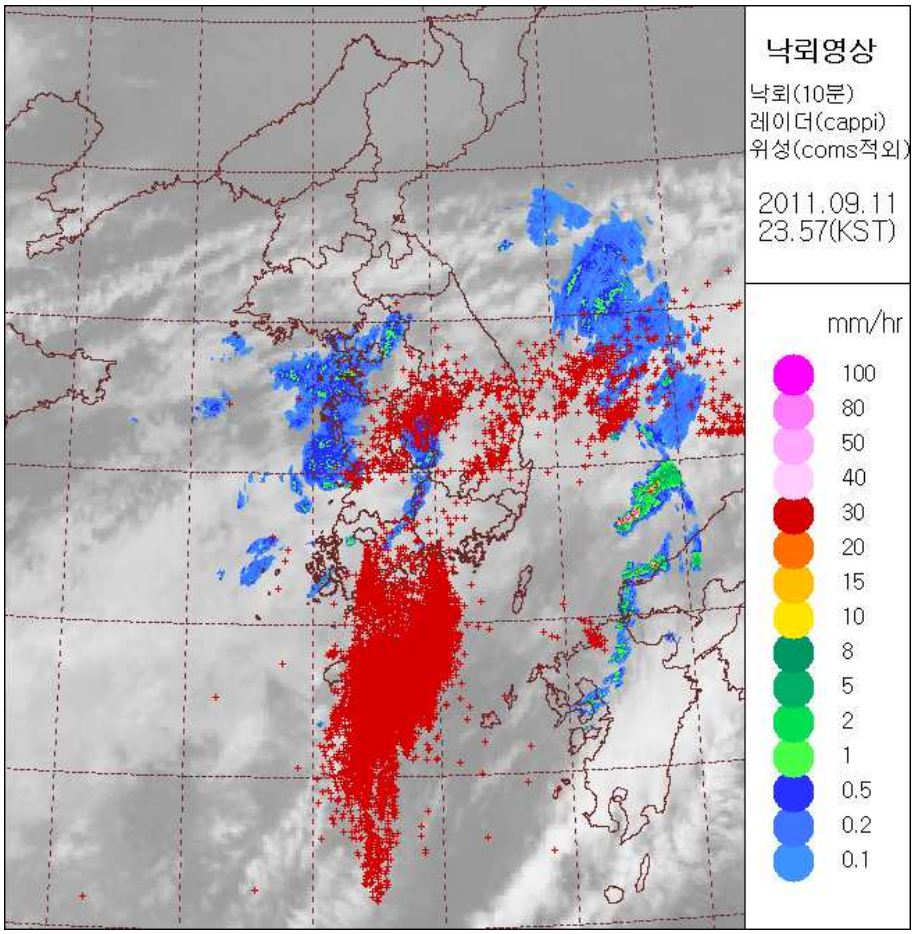
o 위 성 : 1024 * 1024 * 2bytes(2km격자 COMS 적외)

- 격자에 해당되는 위·경도자료가 Mask화일로 저장

o 낙 뢰 : Text

- 자료에 위,경도가 포함

6.2 이미지 변환



6.3 변환 소스

```
/*************************************************************************
 * project : 낙뢰+ 레이더_위성 합성                                      *
 * comfile : Makefile 참조                                               *
 *************************************************************************/
#include <string.h>
#include <stdlib.h>
#include <gd.h>
#include <math.h>
#include "comp.h"

gdImagePtr im;
char YYMM[100];

int YY, MM, DD, HH, MI;
int color[254];
short **DATA;
short **SDATA[SYDIM][SXDIM];

int main(int argc, char *argv[])
{
        FILE *pngout;
        char str[255];
        char imsi[100]="";
        char *f = "../font/times.ttf";
        char YYMM[100]="";
        int i, j, k;

        char infile[100]="";
        char outfile[100]="";
        char tmp[100]="";

        im = gdImageCreate(ImgX+ XOFF, ImgY);

        //칼라 테이블 읽기
        read_color(im, color);

        //echo box 배경색
        gdImageFilledRectangle(im, 0, 0, ImgX, ImgY, color[20]);

        DATA = (short **)malloc((unsigned) (RYDIM + 1)*sizeof(short*));

        //sat 그리기
        sprintf(tmp,"%s",argv[2]);
        read_sat( im, color, tmp );

        //echo 그리기
        sprintf(tmp,"%s",argv[1]);
        read_radar( im, color, tmp, 0 );

        //위경도 그리기
        latlon_draw( im, color );

        //AWS지명 넣기
        //aws_name( im, color );

        //지도 그리기
        map_draw( im, color );

        //lgt 그리기
        sprintf(tmp,"%s",argv[3]);
        read_light( im, color, tmp );
```

```
        //colorbar 그리기
        title_draw( im, color );

        gdImageRectangle(im, 0,    0,     ImgX,          ImgY-1, color[33]);  // echobox 테두리선
        gdImageRectangle(im, ImgX,0,     ImgX+XOFF-1, ImgY-1, color[33]);  // titlebox 테두리선
        gdImageRectangle(im, ImgX,YOFF, ImgX+XOFF-1, ImgY-1, color[33]);  // titlebox 중간선

    free(DATA);

    strncpy(infile, argv[1], strlen(argv[1])-4-strlen(strrchr(argv[1], '.')));
    sprintf(outfile, "%s.png",infile);
    printf("Outfile  : %s ₩n",outfile);
    pngout  = fopen(outfile,"wb");
    gdImagePng(im,pngout);
    fclose(pngout);
    gdImageDestroy(im);

}
```

```
//Makefile
bin = .
map_lib = ../gis/lamcproj.c
fun_lib = ./function.c

INCLUDEDIRS = -I./include -I/usr/local/trmm/GVBOX/include
LIBDIRS = -L/usr/local/lib -L/usr/local/trmm/GVBOX/lib

LIBS = -lm -lgd -lpng -lz -lfreetype

rdr_sat_lgt: rdr_sat_lgt.c
        gcc  $(INCLUDEDIRS)  -o  $(bin)/rdr_sat_lgt  rdr_sat_lgt.c  $(map_lib)  $(fun_lib)
$(LIBDIRS) $(LIBS)

clean:
        rm -f $(bin)/rdr_sat_lgt
```

```
//comp.h
#include <stdio.h>
#include <gdfontt.h>
#include <gdfonts.h>
#include <gdfontmb.h>
#include <gdfontl.h>
#include <gdfontg.h>
#include <zlib.h>
#include <zconf.h>
#include "../gis/map_ini.h"

//radar data size
#define  RXDIM  960
#define  RYDIM  1200

//CMOS data size
#define  SXDIM  1024
#define  SYDIM  1024

#define  ZOOM   1.8
#define  XDIM  (int)(RXDIM/ZOOM)
#define  XOFF  120
```

```
#define   YDIM   (int)(RYDIM/ZOOM)
#define   YOFF   200

//Image size
#define ImgX (int)(RXDIM/ZOOM)
#define ImgY (int)(RYDIM/ZOOM)
//#define ImgX 600
//#define ImgY 720


int read_color( gdImagePtr im, int color[] );
int read_radar( gdImagePtr im, int color[], char *filename, int sat );
int read_light( gdImagePtr im, int color[], char *filename );
int read_sat( gdImagePtr im, int color[], char *filename );
int rdr_kma_cmp_sm();
int rdr_kma_cmp_sm1();
int latlon_draw( gdImagePtr im, int color[] );
int title_draw( gdImagePtr im, int color[] );
int aws_name( gdImagePtr im, int color[] );
int map_draw( gdImagePtr im, int color[] );

static void swap2(void *word)
{
        unsigned char *byte;
        unsigned char temp;
        byte     = (unsigned char *)word;
        temp     = byte[0];
        byte[0] = byte[1];
        byte[1] = temp;
}

static void swap2(void *word);
```

```
//function.c
#include <string.h>
#include <stdlib.h>
#include <gd.h>
#include <math.h>
#include "comp.h"

extern int YY, MM, DD, HH, MI;
extern short DATA[RYDIM][RXDIM];
/*********************************************************
 *          레이더자료 읽어 그리기                                *
 *********************************************************/
int read_radar(gdImagePtr im, int color[], char *filename, int sat)
{
   FILE  *fp1;
   char str[255]="";
   int  i, j, n;
   int cindex;
   char buf[RYDIM][RXDIM];

   sprintf(str, "%s",filename);

   printf("!! read_rdr = %s(%d) ₩n",filename, sat);

   if((fp1=gzopen(str, "r")) == NULL) {
        exit(1);
   }

   gzread(fp1, buf, sizeof(buf));
   gzclose(fp1);

   for (j=0;j<RYDIM;j++)
   {
```

```
        for (i=1;i<RXDIM;i++)
        {
           switch(buf[j][i])
           {
                 case 1 ... 11:
                        cindex = 1;
                        break;
                 case 12 ... 18:
                        cindex = 2;
                        break;
                 case 19 ... 23:
                        cindex = 3;
                        break;
                 case 24 ... 28:
                        cindex = 4;
                        break;
                 case 29 ... 34:
                        cindex = 5;
                        break;
                 case 35 ... 39:
                        cindex = 6;
                        break;
                 case 40 ... 42:
                        cindex = 7;
                        break;
                 case 43 ... 44:
                        cindex = 8;
                        break;
                 case 45:
                        cindex = 9;
                        break;
                 case 46 ... 47:
                        cindex = 10;
                        break;
                 case 48 ... 49:
                        cindex = 11;
                        break;
                 case 50:
                        cindex = 12;
                        break;
                 case 51 ... 54:
                        cindex = 13;
                        break;
                 case  55:
                        cindex = 14;
                        break;
                 case 56 ... 58:
                        cindex = 15;
                        break;
                 case 59 ... 99:
                        cindex = 16;
                        break;
                 case -128 :
                        cindex = 20; //관측반경 외각
                        break;
                 case -127 :
                        cindex = 19; //no echo
                        break;
                 default:
                        cindex = 19;
           }
           if (sat == 1) {
                 gdImageSetPixel(im, i/ZOOM, YDIM-j/ZOOM, color[cindex]);
           }
           else {
                 if (cindex > 0 && cindex < 17) gdImageSetPixel(im, i/ZOOM, YDIM-j/ZOOM, color[cindex]);
           }

        }
}

        return 0;
```

```
}

/*******************************************************
 *          낙뢰자료 읽어 그리기                       *
 *******************************************************/
int read_light(gdImagePtr im, int color[], char *filename)
{
   FILE *fplgt;
   char  type;
   char  temp[120];
   int    yr, mon, day, hr, min, itmp, sensor;
   float sec, lat, lon, imp, ftmp;
   float lgtTime, eTime, sTime;

   int brect[8];
   float alon, alat, x2, y2;
   struct lamc_parameter map;
   char *f = "../font/gulim.ttc";

   map.Re = 6370.19584;    // 사용할 지구반경 [ km ]
   map.grid = 2.0;          // 격자간격          [ km ]
   map.slat1 = 30.0;        // 표준위도          [degree]
   map.slat2 = 60.0;        // 표준위도          [degree]
   map.olat = 38.0;        // 기준점의 경도    [degree]
   map.olon = 126.0;        // 기준점의 위도    [degree]
   map.xo = 400/ZOOM;       // 400 기준점의 X좌표  [격자거리]
   map.yo = 789/ZOOM;       // 789 기준점의 Y좌표  [격자거리]
   map.first = 0;           // 시작여부 (0 = 시작)

   sprintf(temp, "%s",filename);
   printf("!! read_lgt = %s ₩n",filename);

   if((fplgt=fopen(temp, "r")) == NULL) {
     exit(1);
   }


   while(fscanf(fplgt, "%d/%d/%d %d:%d:%f %f %f %f %d %f %f %f %d %f %d %c₩n",
      &mon, &day, &yr, &hr, &min, &sec, &lat, &lon, &imp, &itmp, &ftmp, &ftmp, &ftmp, &itmp, &ftmp, &sensor, &type) != EOF)
   {

         alon=lon;
         alat=lat;
         lamcproj(&alon, &alat, &x2, &y2, 0, map);
         if (x2 > 0 && x2 < XDIM && y2 > 0 && y2 < YDIM && sensor > 2)
         {
            //gdImageStringFT(im, &brect[0], color[11], f, 10.0, 0.0,60+(int)x2 ,60+YDIM-(int)y2, "+");
            gdImageLine(im, (int)x2-2 ,YDIM-(int)y2,(int)x2+2 ,YDIM-(int)y2,    color[11]);
            gdImageLine(im, (int)x2    ,YDIM-(int)y2-2,(int)x2 ,YDIM-(int)y2+2, color[11]);
         }
   }

   YY=yr + 2000;
   MM=mon;
   DD=day;
   HH=hr;
   MI=min;

   fclose(fplgt);

   return 0;

}

/*******************************************************
 *          위성 자료 읽어 그리기                      *
 *******************************************************/
int read_sat(gdImagePtr im, int color[], char *filename)
{
   FILE    *fpsat, *fpmap;
   int i, j;
```

```
    int n, cindex;
    int rgb;
    short   sbuf[1024][1024];
    char str[255];
    char s_file[250];

    //============================================================
    int xx, yy;
    float xlat, ylon;
    float klat[1024][1024];
    float klon[1024][1024];

    float alon, alat, x2, y2;
    struct lamc_parameter map;

    map.Re = 6370.19584;    // 사용할 지구반경 [ km ]
    map.grid = 2.0;          // 격자간격          [ km ]
    map.slat1 = 30.0;        // 표준위도          [degree]
    map.slat2 = 60.0;        // 표준위도          [degree]
    map.olat = 38.0;        // 기준점의 경도    [degree]
    map.olon = 126.0;        // 기준점의 위도    [degree]
    map.xo = 400/ZOOM;       // 400 기준점의 X좌표   [격자거리]
    map.yo = 789/ZOOM;       // 789 기준점의 Y좌표   [격자거리]
    map.first = 0;            // 시작여부 (0 = 시작)

     printf("!! read_sat = %s ₩n",filename);

    fpmap = fopen ("./korea_latlon.txt" , "r");
    while (fgets(str, 255, fpmap) != NULL)
    {
          sscanf(str, "%d %d %f %f ", &xx, &yy, &xlat, &ylon );
          klat[yy][xx] = xlat;
          klon[yy][xx] = ylon;
    }
    fclose(fpmap);

    sprintf(s_file, "%s",filename);
    if((fpsat = gzopen(s_file,"rb")) != NULL) {

      gzread(fpsat, sbuf, sizeof(sbuf));
      gzclose(fpsat);

      for (j=0;j<SYDIM;j++)
      {
          for (i=0;i<SXDIM;i++)
          {
              swap2(&sbuf[j][i]);

              alat = klat[j][i];
              alon = klon[j][i];
              lamcproj(&alon, &alat, &x2, &y2, 0, map); //위경도를 좌표로

              rgb=sbuf[j][i]/4;
              if (rgb > 255)  rgb=255;
              if (rgb < 0)  rgb=0;

              if (x2 > 0 && x2 < XDIM && y2 > 0 && y2 < YDIM) //if (x2 > -60 && x2 < 60+XDIM && y2 > -60 && y2 < 60+YDIM)
              {
                  gdImageSetPixel(im, (int)x2, YDIM-(int)y2, color[rgb-60]);
              }

          }
      }
    }
    else {
          int brect[8];
          char *f = "../font/gulim.ttc";
          sprintf(str,"Not found %s",s_file );
          gdImageStringFT(im, &brect[0], color[33], f, 10.0, 0.0,10 ,YDIM-10, str);
    }
    return 0;
}
```

```
/*============================================================================*
 *  Grid data Smoothing (by 1-2-1 Smoothing)
 *============================================================================*/
int rdr_kma_cmp_sm()
{
   float e[4], e1, e2;
   int     sm_num = 2;                  /* Smoothing 횟수 (default:1) */
   int     i, j, k;

   for (k=0; k<sm_num; k++)
   {
         for (j=0; j<=RYDIM-1; j++)
         {
                  e1 = DATA[j][0];
                  e[0] = DATA[j][0];
                  e[1] = DATA[j][1];

                  for (i=1; i<RXDIM; i++)
                  {
                           e[2] = DATA[j][i+1];

                           if (e[0] > -1 && e[1] > -1 && e[2] > -1)
                                    e2 = (e[0] + 2.0*e[1] + e[2]) * 0.25;
                           else if (e[0] > -1 && e[1] <= -1 && e[2] > -1)
                                    e2 = (e[0] + e[2]) * 0.5;
                           else
                                    e2 = e[1];

                           DATA[j][i-1] = (short)e1;
                           e1 = e2;
                           e[0] = e[1];
                           e[1] = e[2];
                  }
                  DATA[j][i-1] = (short)e1;
         }

         for (i=0; i<=RXDIM-1; i++)
         {
                  e1 = DATA[0][i];
                  e[0] = DATA[0][i];
                  e[1] = DATA[1][i];

                  for (j=1; j<RYDIM; j++)
                  {
                           e[2] = DATA[j+1][i];

                           if (e[0] > -1 && e[1] > -1 && e[2] > -1)
                                    e2 = (e[0] + 2.0*e[1] + e[2]) * 0.25;
                           else if (e[0] > -1 && e[1] <= -1 && e[2] > -1)
                                    e2 = (e[0] + e[2]) * 0.5;
                           else
                                    e2 = e[1];

                           DATA[j-1][i] = (short)e1;
                           e1 = e2;
                           e[0] = e[1];
                           e[1] = e[2];
                  }
                  DATA[j-1][i] = (short)e1;
         }
   }
   return 0;
}

/*============================================================================*
* smooth
*============================================================================*/
int rdr_kma_cmp_sm1()
{
  int     i, j;
```

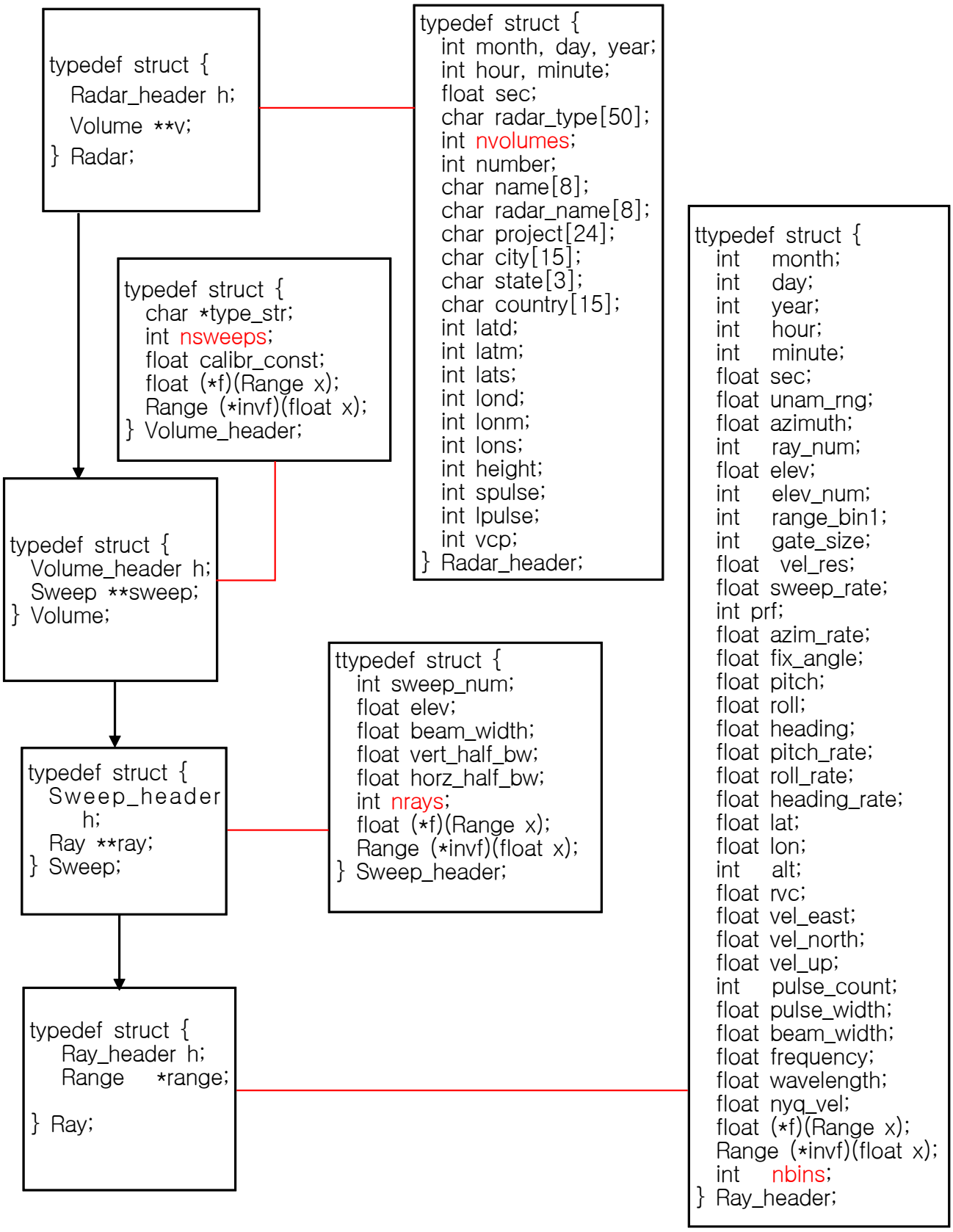
```
  for (j=1; j<RYDIM-1; j++)
  {
     for (i=1; i<RXDIM-1; i++)
     {
         if ( DATA[j][i-1] > 0 && DATA[j][i] == 0 && DATA[j][i+1] > 0) DATA[j][i] = (DATA[j][i-1] + DATA[j][i+1])/2;
     }
  }

  for (i=1; i<RXDIM-1; i++)
  {
     for (j=1; j<RYDIM-1; j++)
     {
         if ( DATA[j-1][i] > 0 && DATA[j][i] == 0 && DATA[j+1][i] > 0) DATA[j][i] = (DATA[j-1][i] +  DATA[j+1][i])/2;
     }
  }

  return 0;
}

/*****************************************************
 *    color table load                                *
 *****************************************************/
int read_color(gdImagePtr im, int    color[])
{
        color[0] = gdImageColorAllocate(im, 110, 190, 255);
        color[1] = gdImageColorAllocate(im, 62, 148, 255);
        color[2] = gdImageColorAllocate(im, 62, 116, 255);
        color[3] = gdImageColorAllocate(im, 40, 49, 255);
        color[4] = gdImageColorAllocate(im, 70, 255, 70);
        color[5] = gdImageColorAllocate(im, 0, 224, 80);
        color[6] = gdImageColorAllocate(im, 0, 175, 101);
        color[7] = gdImageColorAllocate(im, 0, 150, 101);
        color[8] = gdImageColorAllocate(im, 255, 229, 0);
        color[9] = gdImageColorAllocate(im, 255, 190, 0);
        color[10] = gdImageColorAllocate(im, 255, 112, 0);
        color[11] = gdImageColorAllocate(im, 215, 0, 0);
        color[12] = gdImageColorAllocate(im, 255, 206, 255);
        color[13] = gdImageColorAllocate(im, 255, 170, 255);
        color[14] = gdImageColorAllocate(im, 255, 125, 250);
        color[15] = gdImageColorAllocate(im, 255,  0, 255);
        color[16] = gdImageColorAllocate(im, 136, 136, 136);
        color[17] = gdImageColorAllocate(im, 163, 163, 163);
        color[18] = gdImageColorAllocate(im, 110, 190, 255);
        color[19] = gdImageColorAllocate(im, 224, 224, 224); //no echo
        color[20] = gdImageColorAllocate(im, 196, 196, 196); //레인지 외곽
        color[21] = gdImageColorAllocate(im, 255, 255, 255);
        color[22] = gdImageColorAllocate(im, 0, 0, 0);

        color[33] = gdImageColorAllocate(im, 0, 0, 0);
        color[34] = gdImageColorAllocate(im, 149, 149, 149);
        color[35] = gdImageColorAllocate(im, 196, 196, 196);
        color[36] = gdImageColorAllocate(im, 224, 224, 224);
        color[37] = gdImageColorAllocate(im, 255, 255, 255);
        color[38] = gdImageColorAllocate(im, 110,  40,  40);

        int k;
        for (k=40;k<165 ;k++ )
        {
                color[k] = gdImageColorAllocate(im, k+90, k+90, k+90);
        }
        return 0;
}

/*********************************************
* 경도, 위도선 그리기                        *
*********************************************/
int latlon_draw( gdImagePtr im, int color[])
{
        int     lon, lat;
        float   x1, y1;

        float alon, alat, x2, y2;
```

```
        struct lamc_parameter map;

        map.Re = 6370.19584;
        map.grid = 2.0;
        map.slat1 = 30.0;
        map.slat2 = 60.0;
        map.olat = 38.0;
        map.olon = 126.0;
        map.xo = 400/ZOOM;
        map.yo = 789/ZOOM;
        map.first = 0;


        // 경도선 /
        for (lon = 120; lon < 140; lon += 2)
        {
                alon = (float)lon;
                alat = 80;
                lamcproj(&alon, &alat, &x1, &y1, 0, map);

                alon = (float)lon;
                alat = 0;
                lamcproj(&alon, &alat, &x2, &y2, 0, map);
                gdImageDashedLine(im, (int)x1, YDIM-(int)y1, (int)x2, YDIM-(int)y2, color[38]);
        }

        // 위도선 /
        for (lat = 30; lat < 50; lat += 2)
        {
                for (lon = 120*10; lon < 140*10; lon += 1)
                {
                        alon = (float)lon*0.1;
                        alat = (float)lat;
                        lamcproj(&alon, &alat, &x1, &y1, 0, map);

                        alon = (float)lon*0.1 +  1*0.05;
                        alat = (float)lat;
                        lamcproj(&alon, &alat, &x2, &y2, 0, map);
                        gdImageLine(im, (int)x1, YDIM-(int)y1, (int)x2, YDIM-(int)y2, color[38]);
                }
        }

        return 0;
}

/*********************************************************
*       AWS지명 그리기                                   *
*********************************************************/
int aws_name( gdImagePtr im, int color[])
{
   FILE *fpAws;
   char aws_name[50];
   int i;
   int num_aws;
   int brect[8];
   float aws_lat, aws_lon;
   char *f = "../font/gulim.ttc";

   float alon, alat, x2, y2;
   struct lamc_parameter map;

   map.Re = 6370.19584;
   map.grid = 2.0;
   map.slat1 = 30.0;
   map.slat2 = 60.0;
   map.olat = 38.0;
   map.olon = 126.0;
   map.xo = 400/ZOOM;
   map.yo = 789/ZOOM;
   map.first = 0;

   fpAws = fopen("../aws/aws_pos_han_2.txt","r");
```

```
    fscanf(fpAws,"%d",&num_aws);

    for (i = 0; i < num_aws ; i++)
    {
        fscanf(fpAws,"%s %f %f",aws_name,&aws_lat,&aws_lon);
        alon=aws_lon;
        alat=aws_lat;
        lamcproj(&alon, &alat, &x2, &y2, 0, map);
        if (x2 > -60 && x2 < 60+XDIM && y2 > -60 && y2 < 60+YDIM)
        {
            gdImageStringFT(im, &brect[0], color[38], f, 10.0, 0.0,60+(int)x2 ,60+YDIM-(int)y2, aws_name);
        }
    }
    fclose(fpAws);

    return 0;
}

/*********************************************************
*          지도 그리기                                   *
*********************************************************/
int map_draw( gdImagePtr im, int color[])
{
    FILE *fpMap;
    char str[255];
    int line, type;
    int i, j;
    float aws_lat, aws_lon;
    float _x=0, _y=0;

    float alon, alat, x2, y2;
    struct lamc_parameter map;

    map.Re = 6370.19584;
    map.grid = 2.0;
    map.slat1 = 30.0;
    map.slat2 = 60.0;
    map.olat = 38.0;
    map.olon = 126.0;
    map.xo = 400/ZOOM;
    map.yo = 789/ZOOM;
    map.first = 0;

    fpMap = fopen ("../gis/map.txt" , "r");

    while (fgets(str, 1024, fpMap) != NULL)
    {
        j = 0;
        sscanf(str, "%d %d", &line, &type);

        fgets(str, 1024, fpMap);
        sscanf(str, "%f %f", &aws_lon, &aws_lat);
        alon=aws_lon;
        alat=aws_lat;
        lamcproj(&alon, &alat, &x2, &y2, 0, map);
        _x = x2;
        _y = y2;

        for (i=0; i<line - 1; i++)
        {
            fgets(str, 1024, fpMap);

            if (type != 3)
            {
                sscanf(str, "%f %f", &aws_lon, &aws_lat);
                alon=aws_lon;
                alat=aws_lat;
                lamcproj(&alon, &alat, &x2, &y2, 0, map);
                //if (x2 > 0 && x2 < XDIM && y2 > 0 && y2 < YDIM)
                if (abs(x2-_x) < 10 && abs(y2-_y) < 30 && x2 > 0 && x2 < XDIM && y2 > 0 && y2 < YDIM)
                {
```

```
                    gdImageLine(im, (int)(x2+0.5), YDIM-(int)(y2+0.5)-1,
                            (int)(_x+0.5), YDIM-(int)(_y+0.5)-1, color[38]);
            }

            _x = x2;
            _y = y2;
        }
      }
   }
   fclose(fpMap);
   return 0;
}

/*******************************************************
 *          color table 그리기                         *
 *******************************************************/
int title_draw( gdImagePtr im, int color[])
{
   char str[100];
   char *f = "../font/gulim.ttc";
   char *err;
   int i=0;
   int brect[8];

   //colorbar box 배경색
   gdImageFilledRectangle(im, ImgX, 0, ImgX+XOFF, ImgY, color[37]); //37

   FILE *fpTil;
   fpTil = fopen("../color_table/title.txt","r");

   fscanf(fpTil,"%s",str);
   err = gdImageStringTTF(im,&brect[0],color[33], f, 16.0,0.0,ImgX+15 , 40, str);
   err = gdImageStringTTF(im,&brect[0],color[33], f, 16.0,0.0,ImgX+16 , 40, str);

   fscanf(fpTil,"%s",str);
   err = gdImageStringTTF(im,&brect[0],color[33], f, 12.0,0.0,ImgX+3 , 70, str);

   fscanf(fpTil,"%s",str);
   err = gdImageStringTTF(im,&brect[0],color[33], f, 12.0,0.0,ImgX+3 , 90, str);

   fscanf(fpTil,"%s",str);
   err = gdImageStringTTF(im,&brect[0],color[33], f, 12.0,0.0,ImgX+3 , 110, str);
   fclose(fpTil);

   sprintf(str, "%04d.%02d.%02d", YY,MM,DD);
   gdImageStringFT(im, &brect[0], color[33], f, 16.0, 0.0,ImgX+10 , 150, str);
   sprintf(str, "%02d.%02d(KST)", HH, MI);
   gdImageStringFT(im, &brect[0], color[33], f, 16.0, 0.0,ImgX+10 , 170, str);

   static char  rain[][16] ={"0.1","0.2","0.5","1","2","5","8","10","15","20","30","40","50","80","100"," "};
   int xoff, xwith;
   int yoff, ywith;

   for (i=0;i<16 ;i++ )
   {
         xoff  = XOFF/6;
         xwith = XOFF/2;
         ywith = (ImgY-YOFF)/18;
         yoff  = (i+1)*ywith;
         //gdImageFilledRectangle(im, ImgX+xoff, ImgY-yoff, ImgX+xwith, ImgY-ywith-yoff, color[i]);
         if (i > 0) gdImageFilledArc(im, ImgX+xoff+15, ImgY-yoff,30,30,1,360, color[i],0);

         sprintf(str, " %s",rain[i]);
         gdImageStringFT(im, &brect[0], color[22], f, 12.0, 0.0,ImgX+xwith+5 , ImgY-ywith-yoff+5, str);
   }
   sprintf(str, "mm/hr");
   gdImageStringFT(im, &brect[0], color[22], f, 15.0, 0.0,ImgX+xwith ,ImgY-ywith-yoff-5,str);

   return 0;
}
```

[부록1 rsl.h]


typedef struct {
Radar_header h;
Volume **v;
} Radar;
typedef struct {
int month, day, year;
int hour, minute;
float sec;
char radar_type[50];
int nvolumes;
int number;
char name[8];
char radar_name[8];
char project[24];
char city[15];
char state[3];
char country[15];
int latd;
int latm;
int lats;
int lond;
int lonm;
int lons;
int height;
int spulse;
int lpulse;
int vcp;
} Radar_header;
typedef struct {
char *type_str;
int nsweeps;
float calibr_const;
float (*f)(Range x);
Range (*invf)(float x);
} Volume_header;
typedef struct {
Volume_header h;
Sweep **sweep;
} Volume;
ttypedef struct {
int sweep_num;
float elev;
float beam_width;
float vert_half_bw;
float horz_half_bw;
int nrays;
float (*f)(Range x);
Range (*invf)(float x);
} Sweep_header;
typedef struct {
Sweep_header h;
Ray **ray;
} Sweep;
typedef struct {
Ray_header h;
Range *range;
} Ray;
ttypedef struct {
int month;
int day;
int year;
int hour;
int minute;
float sec;
float unam_rng;
float azimuth;
int ray_num;
float elev;
int elev_num;
int range_bin1;
int gate_size;
float vel_res;
float sweep_rate;
int prf;
float azim_rate;
float fix_angle;
float pitch;
float roll;
float heading;
float pitch_rate;
float roll_rate;
float heading_rate;
float lat;
float lon;
int alt;
float rvc;
float vel_east;
float vel_north;
float vel_up;
int pulse_count;
float pulse_width;
float beam_width;
float frequency;
float wavelength;
float nyq_vel;
float (*f)(Range x);
Range (*invf)(float x);
int nbins;
} Ray_header;

초단기 레이더강수량예측시스템 소개

제1장 서론

1.1 배경 및 목적

초단기 강수량 예측을 위하여 레이더 기반의 초단기 강수량 예측모델인 1)VSRF(지구환경시스템연구과 개발)와 2)MAPLE(예보연구과 개발)이 운영되어 왔다. 그러나 두 시스템이 각각 독립적으로 운영되고 있어 사용자들의 활용도가 낮았으며, 관측자료(AWS, 레이더)와 상이한 포맷으로 제공되어 비교분석에 어려움이 있었다. 특히 2010년 6월 15일부터 초단기예보가 실시됨에 따라 초단기 예보를 지원할 수 있는 시스템이 필요하게 되었다. 따라서 따로따로 표출되었던 레이더 강수량 예측자료들을 같은 포맷으로 한 번에 비교분석할 수 있도록 구축하였다.

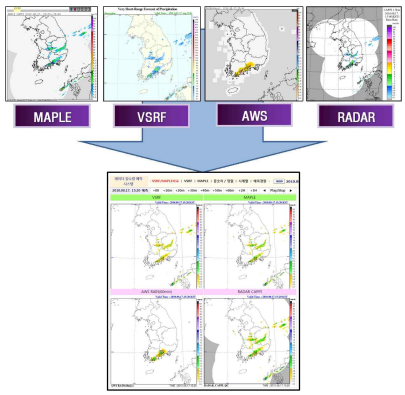


그림 1.1.1 레이더강수량 예측시스템

또한, 레이더 자료 중 예보관들이 가장 많이 활용하는 레이더 강수량 추정값을 문숫자, 시계열 등으로 표출하여, 레이더 영상 표출 형태를 사용자 친화적으로 개선하여 호우특보와 강수량 예측역량향상에 기여하고자 한다.

이 기술노트는 예보관들이 레이더 강수량 예측시스템을 보다 쉽게 사용할 수 있도록 각 구성요소에 대해 설명하였으며, 향후 강수량 예측시스템이나 표출형태를 개선하기 위한 이해를 돕기 위해 작성되었다.

1.2 기술노트 작성내용 범위

예측모델자료, 레이더강수량 예측시스템의 구성요소, 현업운영체계에 대하여 기술하였다.

---

1) VSRF : Very Short Range Forecast
2) MAPLE : McGill Algorithm for Precipitation Nowcasting Using Semi_Lagrangian Extrapolation

2. 사용자료

2.1 예측모델자료

단시간 강수 예보를 목적으로 기상레이더의 에코를 외삽하는 개념은 Ligda (1953)에 의해 처음으로 시도되었고, 그 이후 Noel과 Fleisher (1960), Hilst와 Russo (1960)는 서로 다른 시간에 관측된 2개의 레이더 영상에 대한 상관분석을 실시하여 가장 큰 상관값을 갖는 위치를 에코 패턴 평균 운동의 최적 추정치로서 사용한 후 크기나 강도에 변화가 없는 벡터로 에코장을 외삽 추정했다. 이와 같은 선구적인 연구와 노력에 힘입어서 Rinehart (1981)는 전체 강수패턴을 위해 단순한 평균 벡터를 사용하는 대신에 반사도장 안에서 차등운동(differential motions)을 얻기 위하여 상호상관관계(cross-correlation)를 이용하였고, 독립적인 강우셀의 추적은 National Severe Storms Laboratory (NSSL)의 과학자들에 의하여 1970년대 초기에 시작되었다. 그들은 레이더 반사도 자료와 외삽에 의해 추적된 강수셀의 중심으로부터 대류형 강수셀의 특징을 규명하는 기술을 개발하였다.

국내에서도 단시간에 국지적으로 발생하는 집중호우에 대한 신속한 예측을 위하여 위성 및 레이더 자료를 이용한 단시간 강수예보시스템 개발 연구(기상연구소, 2000), 초단시간 강수특성분석 및 예보모델 개발(기상연구소, 2003) 등의 관련 연구들이 수행되고 있다. 기상연구소에서는 2005년 VSRF, 2007년 MAPLE을 개발하여 실시간 분석 강수량 및 초단시간 예측강수량 자료를 활용할 수 있는 자료를 제공하고 있다.

2.1.1 VSRF 1km

VSRF는 초기장의 패턴상관에 의해 이동벡터를 생산하고 이를 이용하여 시공간적으로 외삽을 하여 6시간 이내의 강수이동을 예측하는 모형이다(국립기상연구소,2005).

- 레이더 입력자료 : 반사도

- 모델영역 : 1 km의 수평분해능을 갖는 800 × 800(800 km × 800 km)
- 활용자료 : 11개 사이트 1.5km CAPPI 합성장(3)RAR)
- 추정기법 : 1km 격자 강수자료 이동벡터 산출기법

① 이동벡터의 후보를 생산하는 단계

⇒ 패턴상관에 의해 1시간동안의 강수의 이동을 계산하고 상관도가 높은 순으로 10개의 후보를 만듦.

② 생산된 10개의 후보들에 이전 시간의 이동벡터와 700hPa RDAPS 바람장을 포함하여 이웃 영역과의 공간 상관성 및 이전 시간 이동벡터와의 시간 상관성을 고려하여 최적의 이동벡터 하나를 결정하는 단계

③ 지형효과에 의한 강수의 증감률을 계산하는 단계

⇒ 지형성 강수분포 및 쇠약에 따라 강수시스템의 발달과 소멸을 고려하여 강수량의 증감을 계산

④ 구해진 이동벡터와 강수량의 증감률을 RAR 강수량에 적용하여 6시간까지의 강수를 예측하는 단계

- 자료주기 및 예측시간 : 10분 간격 1시간, 1시간 간격 6시간 예측
- 수행과정 및 처리시간 : 총 24분

표 2.1.1.1 VSRF 수행내역별 처리시간

모델명	수행 내역	소요시간
VSRF	레이더관측	10분
	VSRF RUN	6분
	수행결과자료수집	1분
	통합표출영역 이미지 생산	5분
	지점별 문·숫자자료 추출	2분

3) RAR : RADAR AWS RAINRATE

2.1.2 MAPLE

MAPLE은 관측된 레이더 반사도 이미지 자료를 이용하여 최적의 이동벡터 산출을 통해 향후 강수에코의 이동을 예측하는 모형이다(국립기상연구소,2007). MAPLE은 반라그랑지안 외삽법을 적용하여 단순선형상관에 의한 단점을 극복하고, 규모별 수명주기를 고려한 확률 예측이 가능하다.

- 레이더 입력자료 : 반사도
- 활용자료 : 11개 사이트 1.5km QC_CAPPI 합성장
- 추정기법 : 2차원 격자형 합성장을 이용한 변분에코추적기법
- 모델영역 : 1 km의 수평분해능을 갖는 1024 × 1024(1024 km × 1024 km)
- 자료주기 및 예측시간 : 10분 간격 3시간
- 수행과정 및 처리시간 : 총 28분

표 2.1.2.1 MAPLE 수행내역별 처리시간

모델명	수행 내역	소요시간
MAPLE	레이더관측	10분
	MAPLE RUN	9분
	수행결과자료수집	2분
	통합표출영역 이미지 생산	5분
	지점별 문·숫자자료 추출	2분

2.2 관측자료

2.2.1 RADAR

레이더 자료는 품질관리(Quality Control)를 거친 1.5km의 QC_CAPPI 자료를 사용하였다.

레이더 자료는 URL API 기반의 자료를 활용하였다. 이 API는 리눅스에 기본 설치된 cURL 라이브러리를 이용하여 C, Fortran에서 웹페이지를 파일처럼 OPEN하여 읽어서 사용할 수 있도록 하는 C 및 Fortran용 서브루린 프로그램으로, 원격지에 있는 자료를 웹 인터페이스를 통하여

쉽게 이쪽 서버의 파일처럼 읽어서 사용할 수 있다는 것과 구현이 쉽고 오픈소스로 되어 있어 수정 및 추가가 쉽다는 장점이 있다.

레이더 자료 URL 예 :

http://172.20.156.33/cgi-bin/url/rdr_cmp_data?tm=201004281230&data=3&qcd=0&mode=1&help=1

표 2.2.1.1 레이더 자료 URL 자료 포맷

변수명	의미	설명
tm	연월일시분 (KST)	합성자료의 시각 (없으면 가장 최근파일의 시간)
data	자료종류	1-CAPPI(1.5km), 2-CMAX, 3-PPI0 (없으면 1)
qcd	QC여부	0-QC전 자료 사용, 1-QC된 자료 사용 (없으면 0)
mode	제공방법	0-자료만, 1-자료+정보, 2-정보만, 3-이진포맷(short=2byte) (없으면 0)
help	도움말추가	1 이면 필드에 대한 약간의 도움말 추가 (0 이거나 없으면 없음)
자료포맷		- 자료의 영역 : 1152km * 1440km - 격자점 수 : 1153 * 1441 - 저장 순서 : 왼쪽아래->오른쪽위, 가로 우선 - 기준점 : (126E, 38N), 왼쪽에서 560km, 아래에서 840km 지점으로 격자점으로는 (561,841)번째 격자점(1~1153개 격자점 중에서 - 아래에 표출되는 순서도 왼쪽아래에서 오른쪽위 순서로 표출되고 있음. - 자료값(dBZ) : -300.00 (레이더영역밖), -250.00 (No Data)

- 레이더 입력자료 : 반사도
- 활용자료 : 11개 사이트 1.5km QC_CAPPI 합성장
- 관측영역 : 1 km의 수평분해능을 갖는 1153 × 1441(1153 km × 1441 km)
- 관측주기 : 10분

2.2.2 AWS

AWS는 매 10분마다 1시간누적강수량을 이용하였으며, AWS 자료도

URL API의 자료를 활용하였다.

국내지점정보 URL 예 :

http://172.20.156.33/url/stn_inf.php?inf=AWS&tm=201007151200&help=1 (AWS)

표 2.2.2.1 국내지점정보 URL 자료 포맷

변수명	의미	설명
inf	지점종류	SFC(지상), AWS(AWS), BUOY(BUOY), AIR(항공), UPP(고층)
tm	년월일시분(KST)	해당 시점에 존재하는 지점목록 (없으면 현재시각)
stn	지점번호	해당 지점의 정보 표출 (0 이거나 없으면 전체지점)
help	도움말추가	1 이면 필드에 대한 약간의 도움말 추가 (0 이거나 없으면 없음)

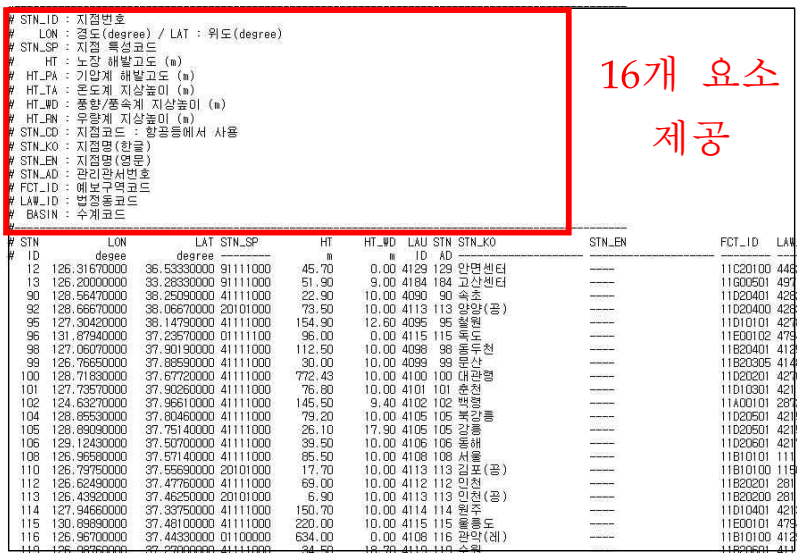


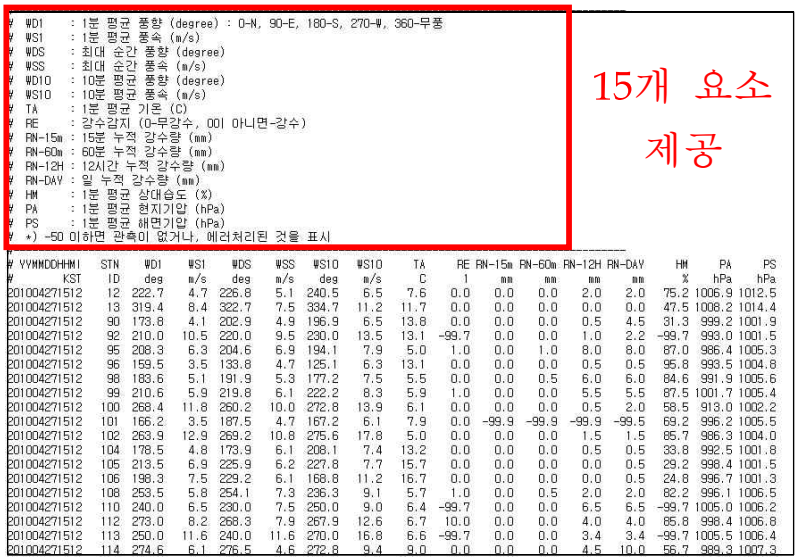
그림 2.2.2.1 국내지점정보 URL 예시

AWS 매분 자료 URL 예 :

http://172.20.156.33/cgi-bin/url/aws_min?tm=201004271512&stn=0&help=1

표 2.2.2.2 AWS 매분자료 URL 자료 포맷

변수명	의미	설명
tm	년월일시분(KST)	해당 시점에 존재하는 지점목록 (없으면 현재시간)
stn	지점번호	해당 지점의 정보 표출 (0 이거나 없으면 전체지점)
help	도움말추가	1 이면 필드에 대한 약간의 도움말 추가 (0 이거나 없으면 없음)




```
#  WD1     : 1분 평균 풍향 (degree) : 0-N, 90-E, 180-S, 270-W, 360-무풍
#  WS1     : 1분 평균 풍속 (m/s)
#  WDS     : 최대 순간 풍향 (degree)
#  WSS     : 최대 순간 풍속 (m/s)
#  WD10    : 10분 평균 풍향 (degree)
#  WS10    : 10분 평균 풍속 (m/s)
#  TA      : 1분 평균 기온 (C)
#  RE      : 강수감지 (0-무강수, 0이 아니면-강수)
#  RN-15m  : 15분 누적 강수량 (mm)
#  RN-60m  : 60분 누적 강수량 (mm)
#  RN-12H  : 12시간 누적 강수량 (mm)
#  RN-DAY  : 일 누적 강수량 (mm)
#  HM      : 1분 평균 상대습도 (%)
#  PA      : 1분 평균 현지기압 (hPa)
#  PS      : 1분 평균 해면기압 (hPa)
#  *) -50 이하면 관측이 없거나, 에러처리된 것을 표시
```

# YYMMDDHHMI	STN	WD1	WS1	WDS	WSS	WS10	TA	RE	RN-15m	RN-60m	RN-12H	RN-DAY	HM	PA	PS
# KST	ID	deg	m/s	deg	m/s	deg	m/s	C	1	mm	mm	mm	mm	%	hPa	hPa
201004271512	12	222.7	4.7	226.8	5.1	240.5	6.5	7.6	0.0	0.0	0.0	2.0	2.0	75.2	1006.9	1012.5
201004271512	13	319.4	8.4	322.7	7.5	334.7	11.2	11.7	0.0	0.0	0.0	0.0	0.0	47.5	1008.2	1014.4
201004271512	90	173.8	4.1	202.9	4.9	196.9	6.5	13.8	0.0	0.0	0.0	0.5	4.5	31.3	999.2	1001.9
201004271512	92	210.0	10.5	220.0	9.5	230.0	13.5	13.1	-99.7	0.0	0.0	1.0	2.2	-99.7	993.0	1001.5
201004271512	95	208.3	6.3	204.6	6.9	194.1	7.9	5.0	1.0	0.0	1.0	8.0	8.0	87.0	986.4	1005.3
201004271512	96	159.5	3.5	133.8	4.7	125.1	6.3	13.1	0.0	0.0	0.0	0.5	0.5	95.8	993.5	1004.8
201004271512	98	183.6	5.1	191.9	5.3	177.2	7.5	5.5	0.0	0.0	0.5	6.0	6.0	84.6	991.9	1005.6
201004271512	99	210.6	5.9	219.8	6.1	222.2	8.3	5.9	1.0	0.0	0.0	5.5	5.5	87.5	1001.7	1005.4
201004271512	100	268.4	11.8	260.2	10.0	272.8	13.9	6.1	0.0	0.0	0.0	0.5	2.0	58.5	913.0	1002.2
201004271512	101	166.2	3.5	187.5	4.7	167.2	6.1	7.9	0.0	-99.9	-99.9	-99.9	-99.5	69.2	996.2	1005.5
201004271512	102	263.9	12.9	269.2	10.8	275.6	17.8	5.0	0.0	0.0	0.0	1.5	1.5	85.7	986.3	1004.0
201004271512	104	178.5	4.8	173.9	6.1	208.1	7.4	13.2	0.0	0.0	0.0	0.5	0.5	33.8	992.5	1001.8
201004271512	105	213.5	6.9	225.9	6.2	227.8	7.7	15.7	0.0	0.0	0.0	0.0	0.5	29.2	998.4	1001.5
201004271512	106	198.3	7.5	229.2	6.1	168.8	11.2	16.7	0.0	0.0	0.0	0.0	0.5	24.8	996.7	1001.3
201004271512	108	253.5	5.8	254.1	7.3	236.3	9.1	5.7	1.0	0.0	0.5	2.0	2.0	82.2	996.1	1006.5
201004271512	110	240.0	6.5	230.0	7.5	250.0	9.0	6.4	-99.7	0.0	0.0	6.5	6.5	-99.7	1005.0	1006.2
201004271512	112	273.0	8.2	268.3	7.9	267.9	12.6	6.7	10.0	0.0	0.0	4.0	4.0	85.8	998.4	1006.8
201004271512	113	250.0	11.6	240.0	11.6	270.0	16.8	6.6	-99.7	0.0	0.0	3.4	3.4	-99.7	1005.5	1006.4
201004271512	114	274.6	6.1	276.5	4.6	272.8	9.4	9.0	0.0	0.0	0.0	4.5	10.0	56.7	989.3	1007.3

그림 2.2.2.2 AWS 매분자료 URL 예시

3. 레이더 강수량 예측시스템

3.1 개요

○ 시스템 URL : http://radar.kma.go.kr/swfs/

○ 표출 위치 : COMIS - 레이더/낙뢰 - 레이더강수량예측

○ 예측 모델 : VSRF, MAPLE

○ 자료제공 주기 및 예측시간 : 매10분 간격으로 3시간 예측

○ 자료제공형태

- 제공지점 : 동네예보 편집지점 및 AWS 680개 지점(첨부1 참조)
- 문숫자/정렬 : AWS 9시간 누적강수량과 예측 강수량 등
- 그래픽 : 1시간누적강우분포, 레이더영상, 예측영상 표출
- 시계열 : 막대와 선 그래프로 누적강수량과 예측 강수량 표출

○ 통합표출 도메인 영역

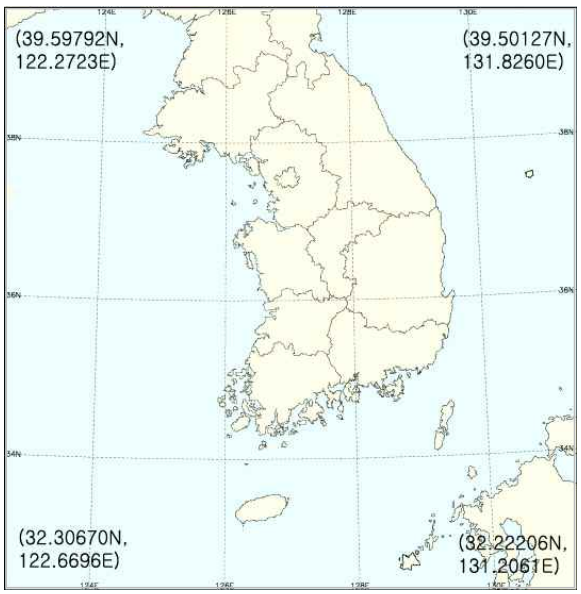


그림 3.1 레이더 강수량 예측시스템
통합표출 도메인 영역 화면

○ 메뉴구성

표 3.1 레이더 강수량 예측시스템 메뉴 구성

1차 메뉴	2차 메뉴	비 고
VSRF/MAPLE 비교	RADAR CAPPI AWS RAIN(60min) VSRF MAPLE	- 강수량분포 및 예측 경향 비교
VSRF	VSRF 예측장 (+10m,+20m,+30m, +40m,+50m,+60m, +2hr, +3hr)	- VSRF 예측자료 표출
MAPLE	MAPLE 예측장 (+10m,+20m,+30m, +40m,+50m,+60m, +2hr, +3hr)	- MAPLE 예측자료 표출
문숫자/정렬	문숫자 - 소속기관별 관할 AWS 예측자료 표출	- AWS 9hr 누적강수량 - VSRF/MAPLE예측강수량 - 12시간 총누적강수량
	정렬 - 전국 및 도단위 총누적강수량 표출	- 전국 및 도 단위 총누적강수량이 많은 지점부터 내림차순 정렬
시계열	- 소속기관별 동네편집지점 시계열 자료	- 막대그래프와 선그래프로 강수량 표출
예측경향	- VSRF/MAPLE - AWS60분강수 - RADAR	- 실황자료 (AWS와 RADAR)와 예측자료와 비교 검증

3.2 시스템 구성

3.2.1 VSRF/MAPLE 비교

VSRF와 MAPLE 모형의 예측 결과를 서로 비교하고 AWS 60분 강우분포도와 레이더 실황 자료를 같은 화면에 표출하여 현재 실황에서 두 모형의 예측 결과을 비교 분석

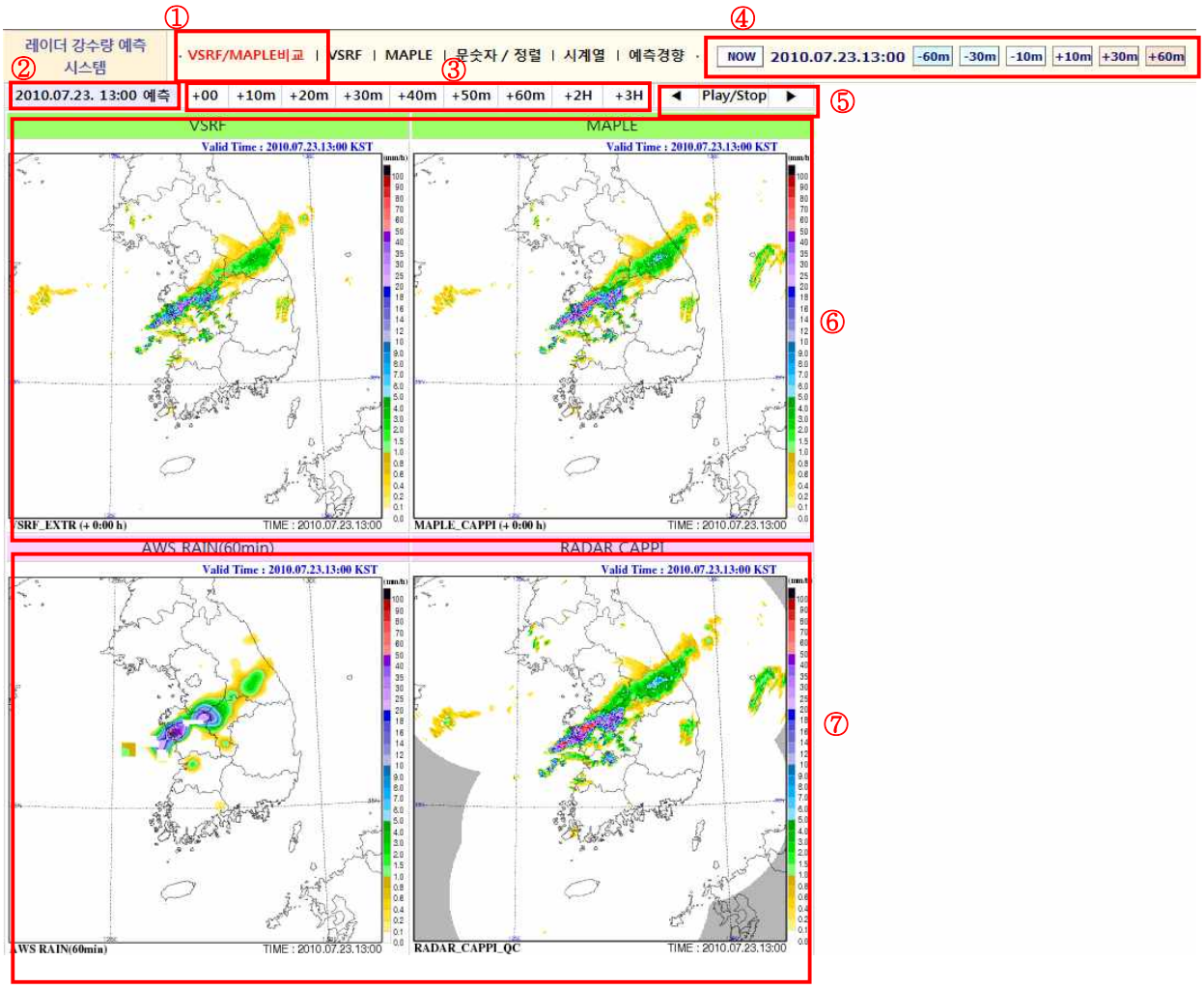


그림 3.2.1 레이더 강수량 예측시스템 VSRF/MAPLE 비교 메뉴 표출 화면

① VSRF/MAPLE 비교 메뉴
② 예측시간
③ 모형 예측 시간 간격
④ 시간 동기화 메뉴
⑤ 예측자료 동영상
⑥ 예측결과 이미지 표출 부분
⑦ 관측자료 표출

3.2.2 VSRF

VSRF 예측결과 자료 표출

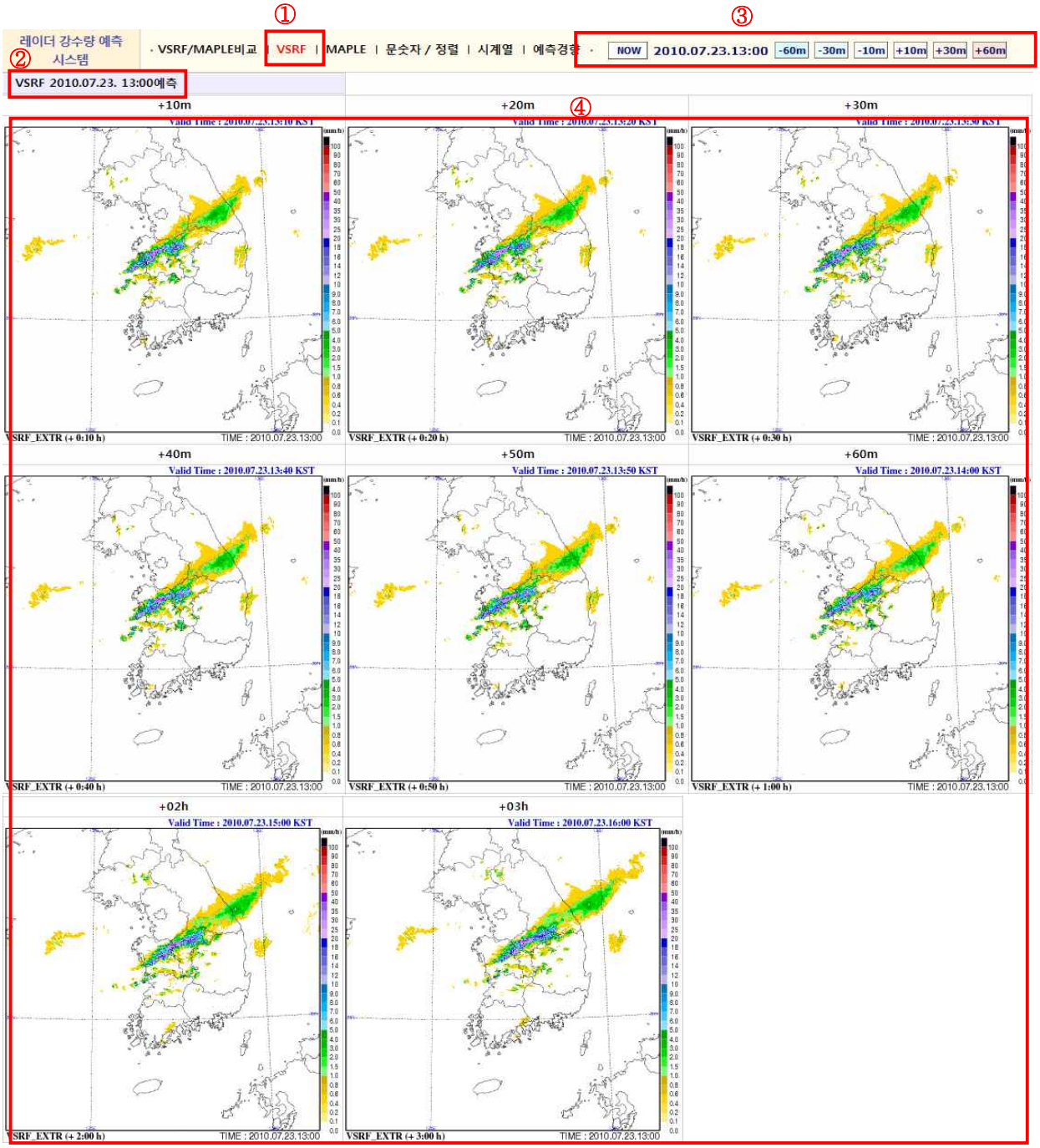


그림 3.2.2 레이더 강수량 예측시스템 VSRF 메뉴 표출 화면

① VSRF 메뉴, ② 예측시간, ③ 시간 동기화 메뉴

④ 예측결과 이미지 표출 부분

3.2.3 MAPLE

MAPLE 예측결과 자료 표출

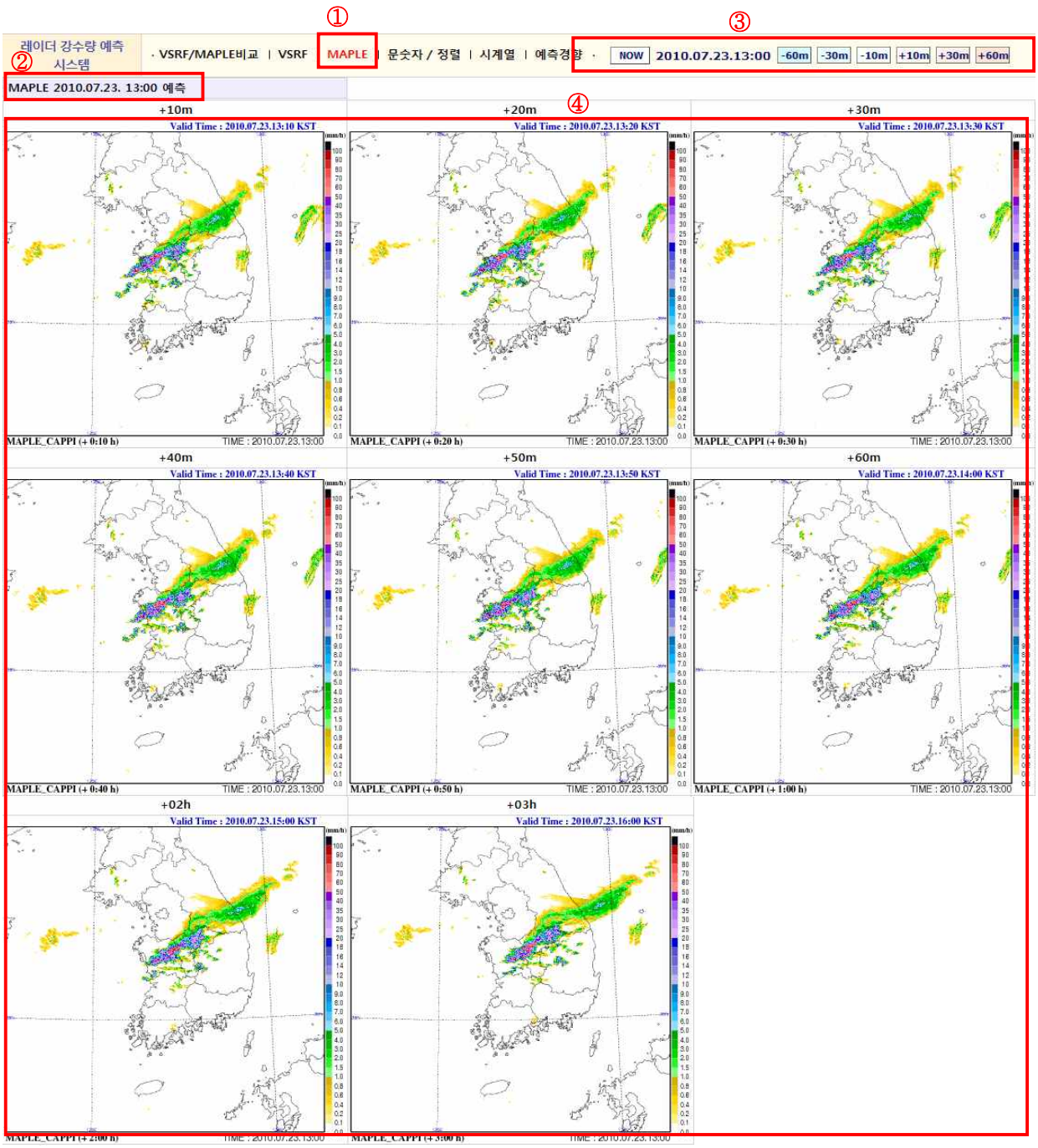


그림 3.2.3 레이더 강수량 예측시스템 MAPLE 메뉴 표출 화면

① MAPLE 메뉴, ② 예측시간, ③ 시간 동기화 메뉴
④ 예측결과 이미지 표출 부분

3.2.4 문숫자

문숫자 자료는 동네예보 편집지점과 AWS 지점에 대한 누적강수량과 모델별 예측자료 및 누적강수량을 제공

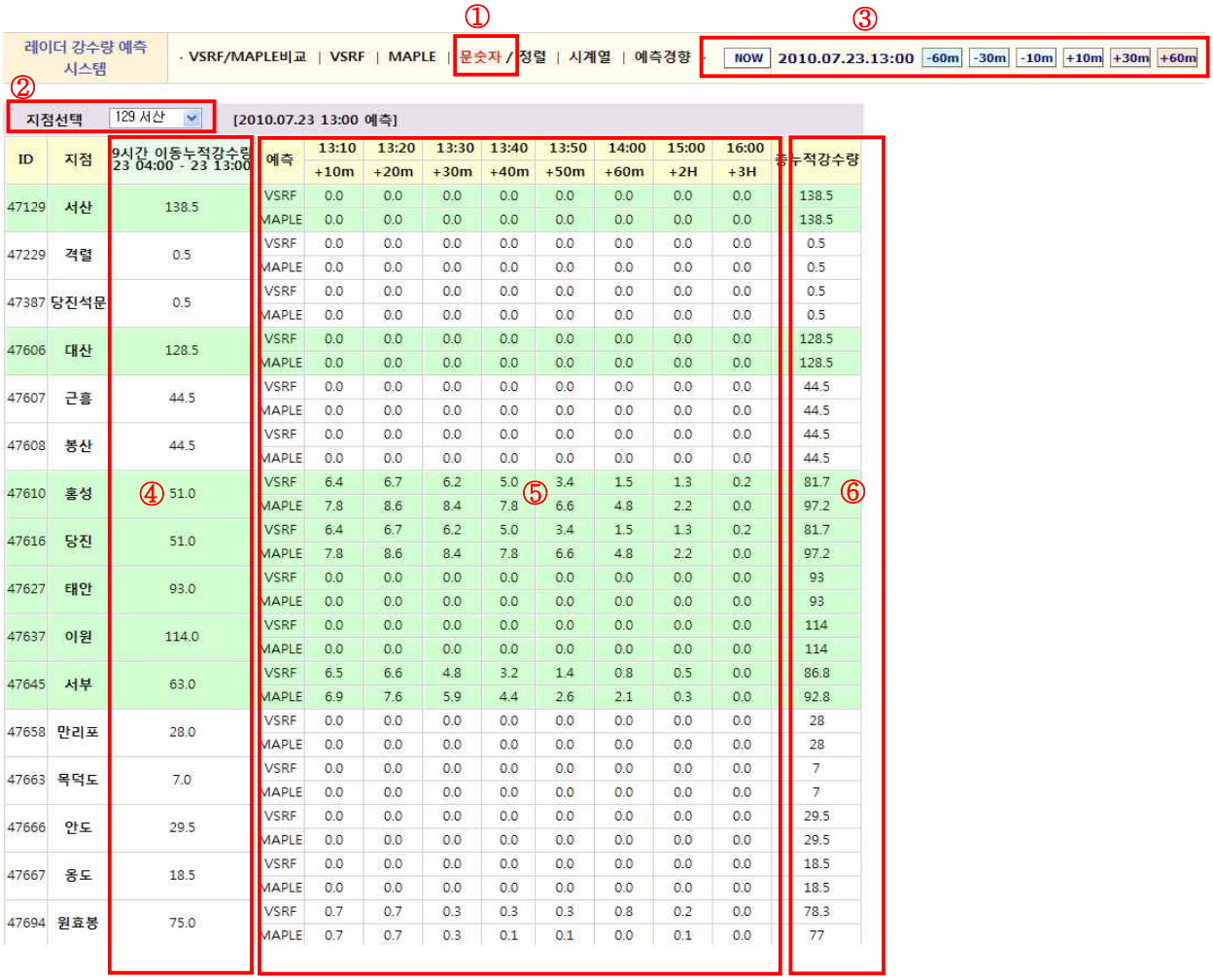
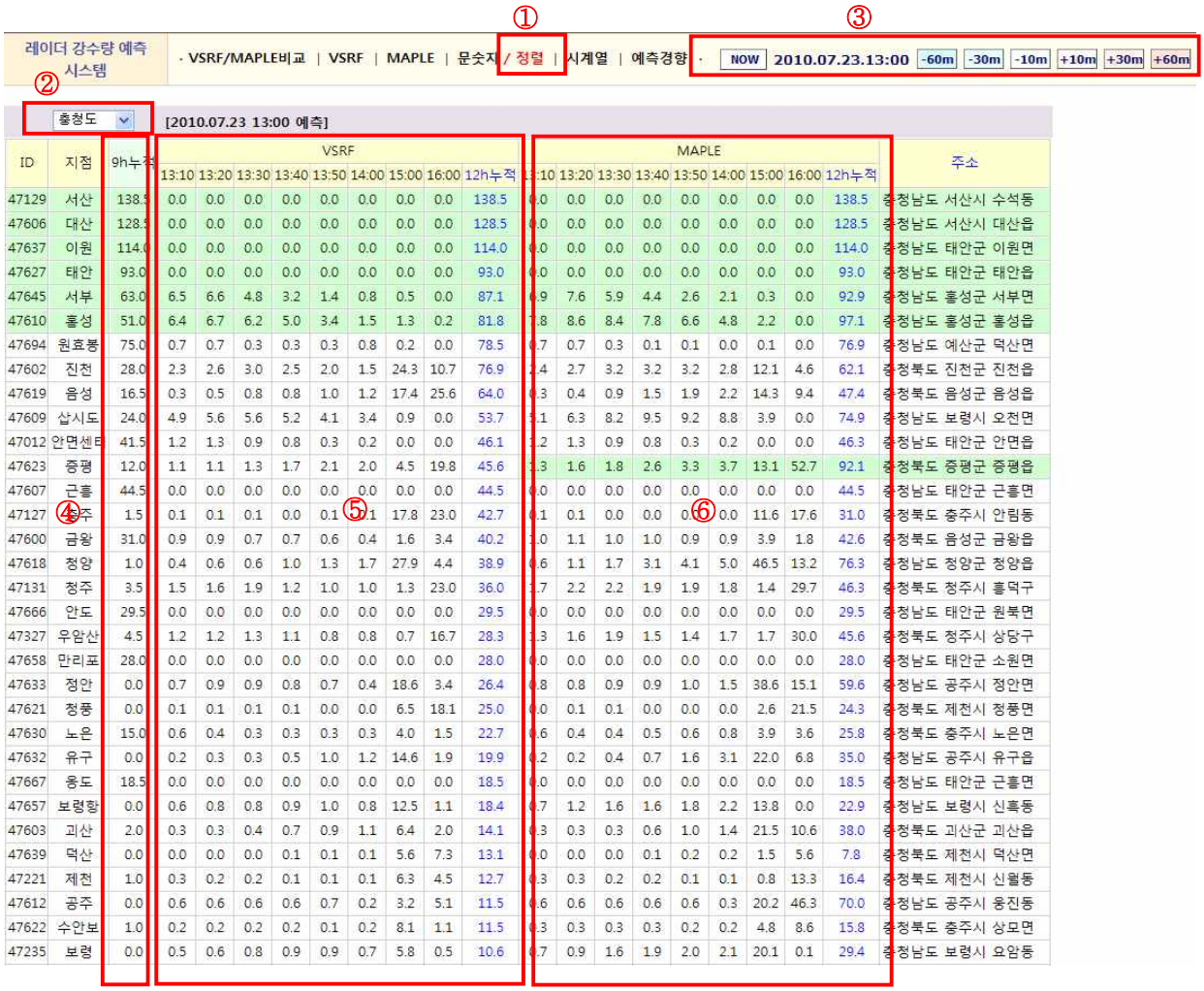


그림 3.2.4 레이더 강수량 예측시스템 문숫자 메뉴 표출 화면

① 문숫자 메뉴
② 지점 선택
③ 시간 동기화 메뉴
④ 지점별 9시간 이동누적강수량
⑤ VSRF와 MAPLE 10분 또는 1시간 예측 강수량
⑥ 지점별 AWS 실황과 예측 강수량을 합한 총 12시간 누적 강수량

3.2.5 문숫자(정렬)

9시간 누적강수량과 모델 예측 강수량의 합을 내림차순으로 정렬하여 가장 강한 강수가 내리는 지점을 손쉽게 파악하도록 구성



ID	지점	9h누적	VSRF 13:10	13:20	13:30	13:40	13:50	14:00	15:00	16:00	12h누적	MAPLE 13:10	13:20	13:30	13:40	13:50	14:00	15:00	16:00	12h누적	주소
47129	서산	138.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	138.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	138.5	충청남도 서산시 수석동
47606	대산	128.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	128.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	128.5	충청남도 서산시 대산읍
47637	이원	114.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	114.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	114.0	충청남도 태안군 이원면
47627	태안	93.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	93.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	93.0	충청남도 태안군 태안읍
47645	서부	63.0	6.5	6.6	4.8	3.2	1.4	0.8	0.5	0.0	87.1	6.9	7.6	5.9	4.4	2.6	2.1	0.3	0.0	92.9	충청남도 홍성군 서부면
47610	홍성	51.0	6.4	6.7	6.2	5.0	3.4	1.5	1.3	0.2	81.8	7.8	8.6	8.4	7.8	6.6	4.8	2.2	0.0	97.1	충청남도 홍성군 홍성읍
47694	원효봉	75.0	0.7	0.7	0.3	0.3	0.3	0.8	0.2	0.0	78.5	0.7	0.7	0.3	0.1	0.1	0.0	0.1	0.0	76.9	충청남도 예산군 덕산면
47602	진천	28.0	2.3	2.6	3.0	2.5	2.0	1.5	24.3	10.7	76.9	2.4	2.7	3.2	3.2	3.2	2.8	12.1	4.6	62.1	충청북도 진천군 진천읍
47619	음성	16.5	0.3	0.5	0.8	0.8	1.0	1.2	17.4	25.6	64.0	0.3	0.4	0.9	1.5	1.9	2.2	14.3	9.4	47.4	충청북도 음성군 음성읍
47609	삽시도	24.0	4.9	5.6	5.6	5.2	4.1	3.4	0.9	0.0	53.7	5.1	6.3	8.2	9.5	9.2	8.8	3.9	0.0	74.9	충청남도 보령시 오천면
47012	안면센터	41.5	1.2	1.3	0.9	0.8	0.3	0.2	0.0	0.0	46.1	1.2	1.3	0.9	0.8	0.3	0.2	0.0	0.0	46.3	충청남도 태안군 안면읍
47623	증평	12.0	1.1	1.1	1.3	1.7	2.1	2.0	4.5	19.8	45.6	1.3	1.6	1.8	2.6	3.3	3.7	13.1	52.7	92.1	충청북도 증평군 증평읍
47607	근흥	44.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	44.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	44.5	충청남도 태안군 근흥면
47127	충주	1.5	0.1	0.1	0.1	0.0	0.1	0.1	17.8	23.0	42.7	0.1	0.1	0.0	0.0	0.0	0.0	11.6	17.6	31.0	충청북도 충주시 안림동
47600	금왕	31.0	0.9	0.9	0.7	0.7	0.6	0.4	1.6	3.4	40.2	1.0	1.1	1.0	1.0	0.9	0.9	3.9	1.8	42.6	충청북도 음성군 금왕읍
47618	청양	1.0	0.4	0.6	0.6	1.0	1.3	1.7	27.9	4.4	38.9	0.6	1.1	1.7	3.1	4.1	5.0	46.5	13.2	76.3	충청남도 청양군 청양읍
47131	청주	3.5	1.5	1.6	1.9	1.2	1.0	1.0	1.3	23.0	36.0	1.7	2.2	2.2	1.9	1.9	1.8	1.4	29.7	46.3	충청북도 청주시 흥덕구
47666	안도	29.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	29.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	29.5	충청남도 태안군 원북면
47327	우암산	4.5	1.2	1.2	1.3	1.1	0.8	0.8	0.7	16.7	28.3	1.3	1.6	1.9	1.5	1.4	1.7	1.7	30.0	45.6	충청북도 청주시 상당구
47658	만리포	28.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	28.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	28.0	충청남도 태안군 소원면
47633	정안	0.0	0.7	0.9	0.9	0.8	0.7	0.4	18.6	3.4	26.4	0.8	0.8	0.9	0.9	1.0	1.5	38.6	15.1	59.6	충청남도 공주시 정안면
47621	청풍	0.0	0.1	0.1	0.1	0.1	0.0	0.0	6.5	18.1	25.0	0.0	0.1	0.1	0.0	0.0	0.0	2.6	21.5	24.3	충청북도 제천시 청풍면
47630	노은	15.0	0.6	0.4	0.3	0.3	0.3	0.3	4.0	1.5	22.7	0.6	0.4	0.4	0.5	0.6	0.8	3.9	3.6	25.8	충청북도 충주시 노은면
47632	유구	0.0	0.2	0.3	0.3	0.5	1.0	1.2	14.6	1.9	19.9	0.2	0.2	0.4	0.7	1.6	3.1	22.0	6.8	35.0	충청남도 공주시 유구읍
47667	웅도	18.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	18.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	18.5	충청남도 태안군 근흥면
47657	보령항	0.0	0.6	0.8	0.8	0.9	1.0	0.8	12.5	1.1	18.4	0.7	1.2	1.6	1.6	1.8	2.2	13.8	0.0	22.9	충청남도 보령시 신흑동
47603	괴산	2.0	0.3	0.3	0.4	0.7	0.9	1.1	6.4	2.0	14.1	0.3	0.3	0.3	0.6	1.0	1.4	21.5	10.6	38.0	충청북도 괴산군 괴산읍
47639	덕산	0.0	0.0	0.0	0.0	0.1	0.1	0.1	5.6	7.3	13.1	0.0	0.0	0.0	0.1	0.2	0.2	1.5	5.6	7.8	충청북도 제천시 덕산면
47221	제천	1.0	0.3	0.2	0.2	0.1	0.1	0.1	6.3	4.5	12.7	0.3	0.3	0.2	0.2	0.1	0.1	0.8	13.3	16.4	충청북도 제천시 신월동
47612	공주	0.0	0.6	0.6	0.6	0.6	0.7	0.2	3.2	5.1	11.5	0.6	0.6	0.6	0.6	0.6	0.3	20.2	46.3	70.0	충청남도 공주시 웅진동
47622	수안보	1.0	0.2	0.2	0.2	0.2	0.1	0.2	8.1	1.1	11.5	0.3	0.3	0.3	0.3	0.2	0.2	4.8	8.6	15.8	충청북도 충주시 상모면
47235	보령	0.0	0.5	0.6	0.8	0.9	0.9	0.7	5.8	0.5	10.6	0.7	0.9	1.6	1.9	2.0	2.1	20.1	0.1	29.4	충청남도 보령시 요암동

그림 3.2.5 레이더 강수량 예측시스템 정렬 메뉴 표출 화면

① 문숫자 정렬 메뉴
② 구역 선택
③ 시간 동기화 메뉴
④ 지점별 9시간 이동누적강수량
⑤ VSRF 예측 강수량과 12시간 누적 강수량
⑥ MAPLE 예측 강수량과 12시간 누적 강수량

3.2.6 시계열

AWS 누적강수량과 예측누적강수량은 선그래프를 이용하였고, 매 10분 예측 강수량은 막대그래프로 강우강도를 표현

그림 3.2.6 레이더 강수량 예측시스템 시계열 메뉴 표출 화면

① 시계열 메뉴
② 지점 선택
③ 시간 동기화 메뉴
④ 10분 또는 1시간 예측 강수량 그래프 축(막대그래프)
⑤ 9시간 누적강수량(선 그래프)
⑥ VSRF/MAPLE 예측 강수량(막대그래프), 누적강수량(선 그래프)
⑦ 총 누적 강수량 그래프 축(선 그래프)

3.2.7 예측경향

AWS 60분 누적강수량 분포도와 RADAR CAPPI 영상과 과거 예측자료를 함께 표출하여 실황과 예측자료의 경향을 분석하도록 구성

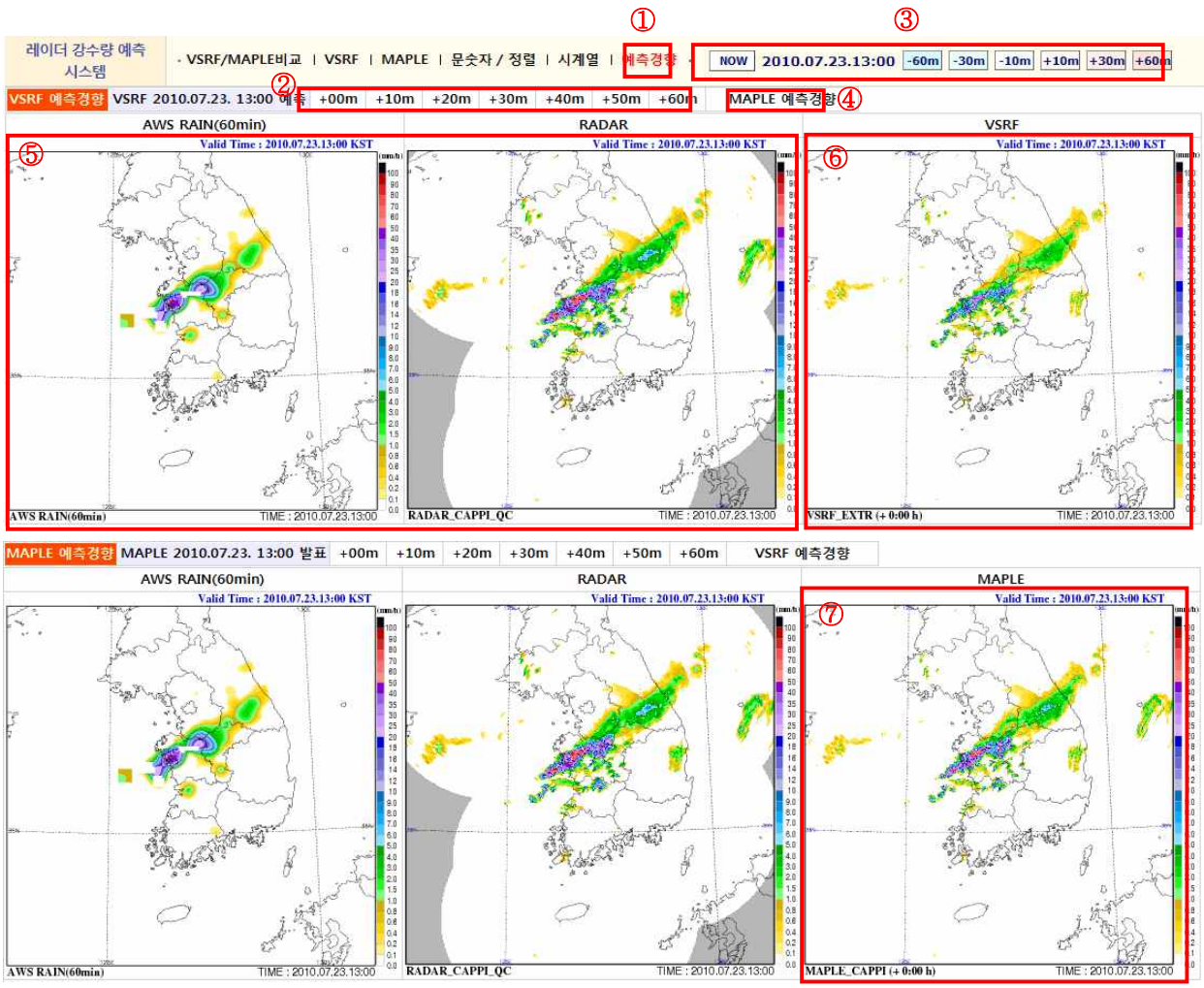


그림 3.2.7 레이더 강수량 예측시스템 예측경향 메뉴 표출 화면

① 검증 메뉴, ② 모형 예측 시간 간격, ③ 시간 동기화 메뉴
④ MAPLE 검증 메뉴, ⑤ 실황(AWS와 레이더)자료
⑥ VSRF 예측장, ⑦ MAPLE 예측장

4. 레이더 강수량 예측성능 검증시스템

4.1 개요

○ 시스템 URL : http://radar.kma.go.kr/swfs/vrfy

○ 표출 위치 : COMIS - 레이더/낙뢰 - 레이더강수량예측/검증

○ 자료제공 형태/주기 : 문숫자 및 그래프 / 10분 간격

○ 성능평가 종류

표 4.1 레이더 강수량 예측성능 검증 항목 및 평가 산출식

항 목	평가 요소	평가 산출식
평균오차(ME)	강수강도(mm/h)	$ME = \frac{1}{N}\sum_{i=1}^{N}(F_i - O_i)$
절대평균오차(MAE)	강수강도(mm/h)	$MAE = \frac{1}{N}\sum_{i=1}^{N}\lvert F_i - O_i \rvert$
평방근오차(RMSE)	강수강도(mm/h)	$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(F_i - O_i)^2}$
상관계수(CORR)	강수유무	$\gamma = \frac{\sum(F-\overline{F})(O-\overline{O})}{\sqrt{\sum(F-\overline{F})^2}\sqrt{\sum(O-\overline{O})^2}}$
정확도(ACC)	강수유무	$ACC = \frac{H+C}{H+M+F+C}$
성공임계지수(CSI)	강수유무	$CSI(\text{or } TS) = \frac{H}{H+M+F}$
편이(Bias)	강수유무	$Bias = (H+F)/(H+M)$

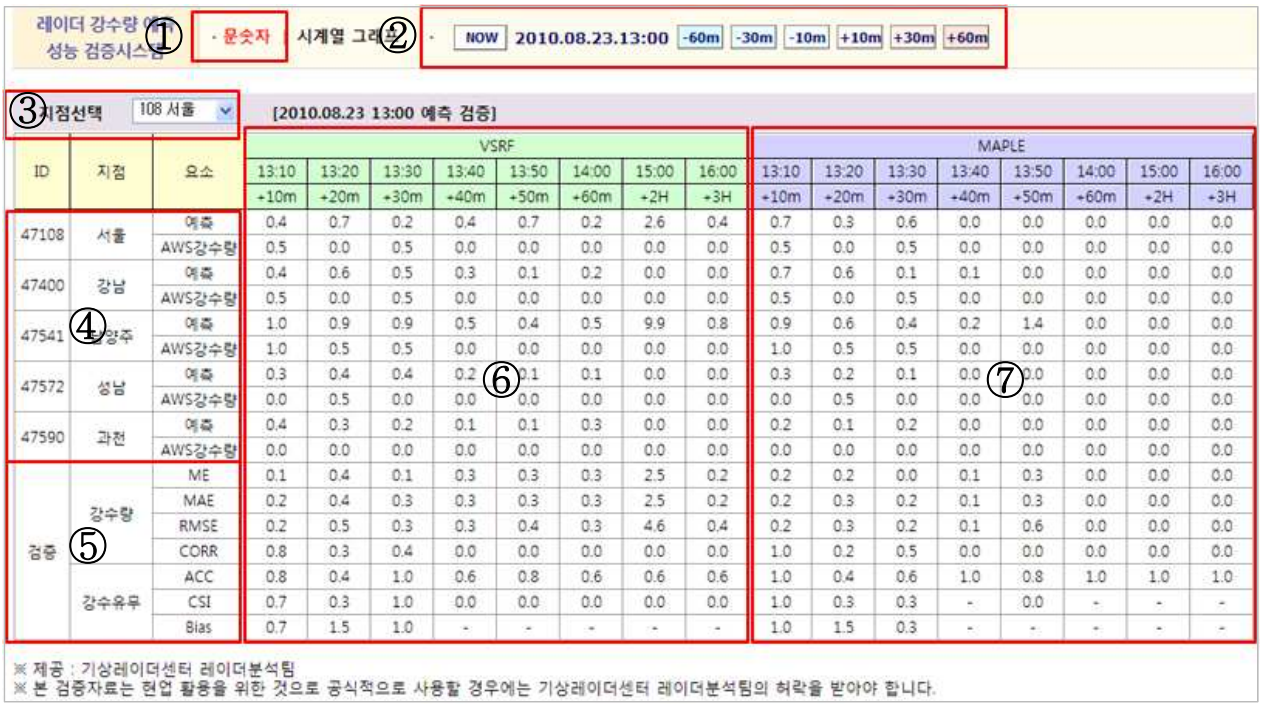
○ 메뉴 구성

1차 메뉴	2차 메뉴	3차 메뉴	내용
문숫자	동네예보 편집지점	VSRF /MAPLE	- AWS 10분 강수량 - 예측 강수량 - ME - MAE - RMSE - CORR - ACC - CSI - Bias
그래프	동네예보 편집지점	VSRF /MAPLE	- ME - RMSE - CSI

4.2 시스템 구성

4.2.1 문숫자

VSRF와 MAPLE 모형에서 예측한 강수량과 AWS 강수량을 지점별 예측과 실황을 서로 비교 할 수 있으며, 강수량과 강수유무에 대한 검증 결과를 각 모형별로 표출



ID	지점	요소	VSRF 13:10 +10m	VSRF 13:20 +20m	VSRF 13:30 +30m	VSRF 13:40 +40m	VSRF 13:50 +50m	VSRF 14:00 +60m	VSRF 15:00 +2H	VSRF 16:00 +3H	MAPLE 13:10 +10m	MAPLE 13:20 +20m	MAPLE 13:30 +30m	MAPLE 13:40 +40m	MAPLE 13:50 +50m	MAPLE 14:00 +60m	MAPLE 15:00 +2H	MAPLE 16:00 +3H
47108	서울	예측	0.4	0.7	0.2	0.4	0.7	0.2	2.6	0.4	0.7	0.3	0.6	0.0	0.0	0.0	0.0	0.0
		AWS강수량	0.5	0.0	0.5	0.0	0.0	0.0	0.0	0.0	0.5	0.0	0.5	0.0	0.0	0.0	0.0	0.0
47400	강남	예측	0.4	0.6	0.5	0.3	0.1	0.2	0.0	0.0	0.7	0.6	0.1	0.1	0.0	0.0	0.0	0.0
		AWS강수량	0.5	0.0	0.5	0.0	0.0	0.0	0.0	0.0	0.5	0.0	0.5	0.0	0.0	0.0	0.0	0.0
47541	남양주	예측	1.0	0.9	0.9	0.5	0.4	0.5	9.9	0.8	0.9	0.6	0.4	0.2	1.4	0.0	0.0	0.0
		AWS강수량	1.0	0.5	0.5	0.0	0.0	0.0	0.0	0.0	1.0	0.5	0.5	0.0	0.0	0.0	0.0	0.0
47572	성남	예측	0.3	0.4	0.4	0.2	0.1	0.1	0.0	0.0	0.3	0.2	0.1	0.0	0.0	0.0	0.0	0.0
		AWS강수량	0.0	0.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.5	0.0	0.0	0.0	0.0	0.0	0.0
47590	과천	예측	0.4	0.3	0.2	0.1	0.1	0.3	0.0	0.0	0.2	0.1	0.2	0.0	0.0	0.0	0.0	0.0
		AWS강수량	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
검증	강수량	ME	0.1	0.4	0.1	0.3	0.3	0.3	2.5	0.2	0.2	0.2	0.0	0.1	0.3	0.0	0.0	0.0
		MAE	0.2	0.4	0.3	0.3	0.3	0.3	2.5	0.2	0.2	0.3	0.2	0.1	0.3	0.0	0.0	0.0
		RMSE	0.2	0.5	0.3	0.3	0.4	0.3	4.6	0.4	0.2	0.3	0.2	0.1	0.6	0.0	0.0	0.0
		CORR	0.8	0.3	0.4	0.0	0.0	0.0	0.0	0.0	1.0	0.2	0.5	0.0	0.0	0.0	0.0	0.0
	강수유무	ACC	0.8	0.4	1.0	0.6	0.8	0.6	0.6	0.6	1.0	0.4	0.6	1.0	0.8	1.0	1.0	1.0
		CSI	0.7	0.3	1.0	0.0	0.0	0.0	0.0	0.0	1.0	0.3	0.3	-	0.0	-	-	-
		Bias	0.7	1.5	1.0	-	-	-	-	-	1.0	1.5	0.3	-	-	-	-	-

그림 4.2.1 레이더 강수량 예측성능 검증시스템 문숫자 자료 표출

① 문숫자 메뉴
② 시간 동기화 메뉴
③ 지점선택
④ 동네예보 지점
⑤ 검증기법 종류
⑥ VSRF 검증자료 표출
⑦ MAPLE 검증자료 표출

4.2.2 시계열 그래프

성공임계지수(CSI)와 평균오차(ME) 및 평방근오차(RMSE)에 대한 검증결과 선 또는 막대그래프로 표출

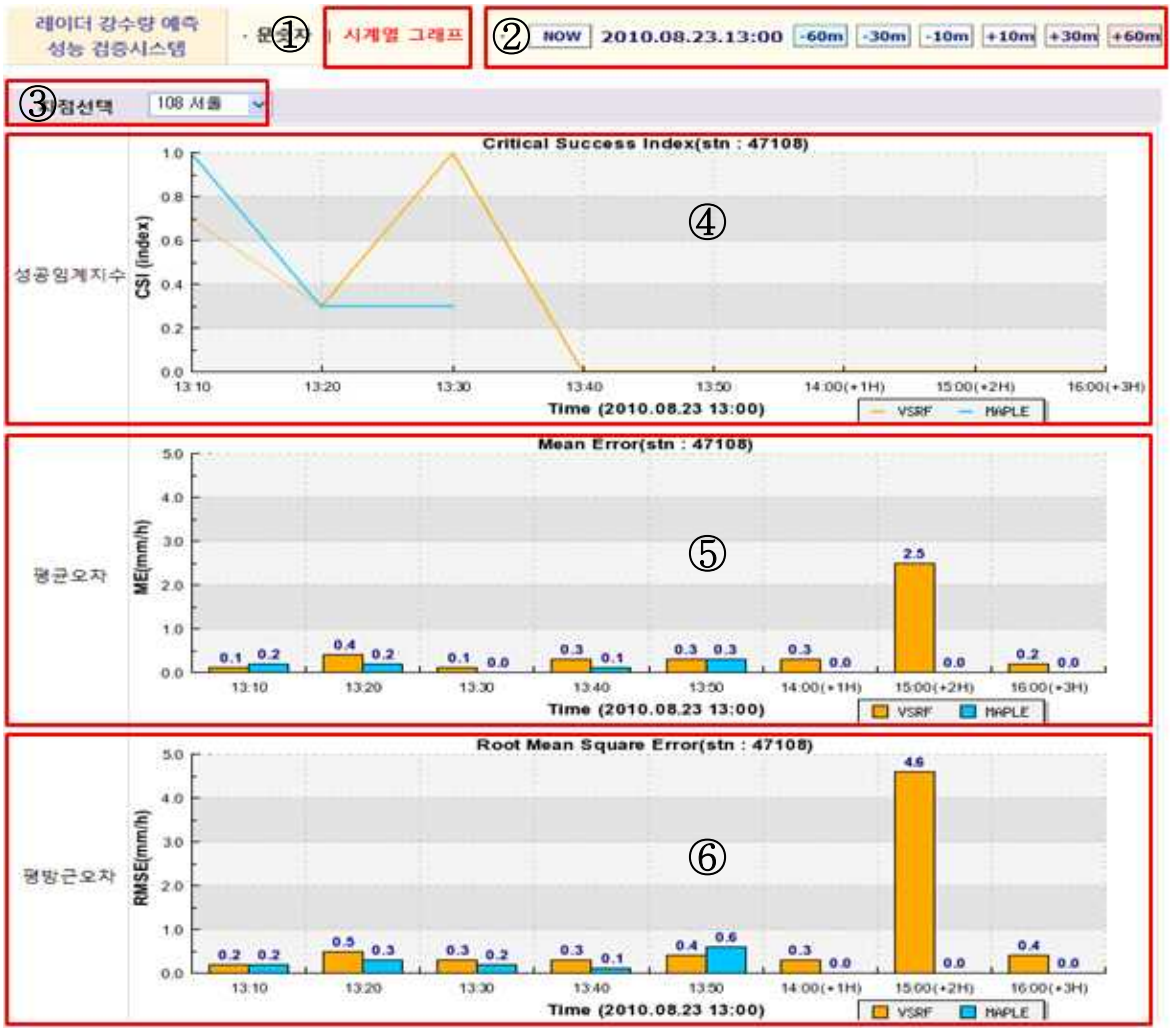


그림 4.2.2 레이더 강수량 예측성능 검증시스템 시계열 그래프 자료 표출

① 시계열 그래프 메뉴
② 시간 동기화 메뉴
③ 지점 선택
④ 성공임계지수 검증자료 표출
⑤ 평균오차 검증자료 표출
⑥ 평방근오차 검증자료 표출