1: motion extraction

```
void ff_estimate_p_frame_motion(MpegEncContext * s,
                                int mb_x, int mb_y)
{
    MotionEstContext * const c= &s->me;
    uint8_t *pix, *ppix;
    int sum, mx, my, dmin;
    int varc;           ///< the variance of the block (sum of squared (p[y][x]-average))
    int vard;           ///< sum of squared differences with the estimated motion vector
    int P[10][2];
    const int shift= 1+s->quarter_sample;
    int mb_type=0;

    FILE *fopen_x;
    FILE *fopen_y;

    --------------

        dmin = ff_epzs_motion_search(s, &mx, &my, P, 0, 0, s->p_mv_table, (1<<16)>>shift, 0, 16);


    fopen_x=fopen("mv_b_x_full.txt","a+");  // Open the output files for motionx
    fopen_y=fopen("mv_b_y_full.txt","a+");  // Open the output files for motiony
    if (s->pict_type==AV_PICTURE_TYPE_P) //Get when get P-frame
    {
    fprintf(fopen_x,"%d\n",mx);  //Outut motion_x
    fprintf(fopen_y,"%d\n",my);  //Outut motion_y
    }

    fclose(fopen_x);
    fclose(fopen_y);
        break;
    }

    /* At this point (mx,my) are full-pell and the relative displacement */
    ppix = c->ref[0][0] + (my * s->linesize) + mx;

    vard = s->mecc.sse[0](NULL, pix, ppix, s->linesize, 16);

    pic->mc_mb_var[s->mb_stride * mb_y + mb_x] = (vard+128)>>8;
    c->mc_mb_var_sum_temp += (vard+128)>>8;

    if (c->avctx->mb_decision > FF_MB_DECISION_SIMPLE) {

        int p_score= FFMIN(vard, varc-500+(s->lambda2>>FF_LAMBDA_SHIFT)*100);
        int i_score= varc-500+(s->lambda2>>FF_LAMBDA_SHIFT)*20;
        c->scene_change_score+= ff_sqrt(p_score) - ff_sqrt(i_score);

        if (vard*2 + 200*256 > varc)
            mb_type|= CANDIDATE_MB_TYPE_INTRA;
```

```
        if (varc*2 + 200*256 > vard || s->qscale > 24){

//        if (varc*2 + 200*256 + 50*(s->lambda2>>FF_LAMBDA_SHIFT) > vard){
            mb_type|= CANDIDATE_MB_TYPE_INTER;
            c->sub_motion_search(s, &mx, &my, dmin, 0, 0, 0, 16);
            if (s->mpv_flags & FF_MPV_FLAG_MV0)
                if(mx || my)
                    mb_type |= CANDIDATE_MB_TYPE_SKIPPED; //FIXME check difference
        }else{
            mx <<=shift;
            my <<=shift;

        }
        if((s->flags&CODEC_FLAG_4MV)
          && !c->skip && varc>50<<8 && vard>10<<8){
            if(h263_mv4_search(s, mx, my, shift) < INT_MAX)
                mb_type|=CANDIDATE_MB_TYPE_INTER4V;

            set_p_mv_tables(s, mx, my, 0);
        }else
            set_p_mv_tables(s, mx, my, 1);
        if((s->flags&CODEC_FLAG_INTERLACED_ME)
          && !c->skip){ //FIXME varc/d checks
            if(interlaced_search(s, 0, s->p_field_mv_table, s->p_field_select_table, mx, my, 0) <
INT_MAX)
                mb_type |= CANDIDATE_MB_TYPE_INTER_I;
        }
    }else{
        int intra_score, i;
        mb_type= CANDIDATE_MB_TYPE_INTER;

        dmin= c->sub_motion_search(s, &mx, &my, dmin, 0, 0, 0, 16);



        if(c->avctx->me_sub_cmp != c->avctx->mb_cmp && !c->skip)
            dmin= get_mb_score(s, mx, my, 0, 0, 0, 16, 1);

        if((s->flags&CODEC_FLAG_4MV)
          && !c->skip && varc>50<<8 && vard>10<<8){
            int dmin4= h263_mv4_search(s, mx, my, shift);

            if(dmin4 < dmin){
                mb_type= CANDIDATE_MB_TYPE_INTER4V;
                dmin=dmin4;
            }
        }
        if((s->flags&CODEC_FLAG_INTERLACED_ME)
          && !c->skip){ //FIXME varc/d checks
            int dmin_i= interlaced_search(s, 0, s->p_field_mv_table, s->p_field_select_table, mx, my,
0);
```

```
            if(dmin_i < dmin){
                mb_type = CANDIDATE_MB_TYPE_INTER_I;
                dmin= dmin_i;
            }
        }

        set_p_mv_tables(s, mx, my, mb_type!=CANDIDATE_MB_TYPE_INTER4V);

    fopen_x=fopen("mv_b_x_sub.txt","a+");
    fopen_y=fopen("mv_b_y_sub.txt","a+");
    if (s->pict_type==AV_PICTURE_TYPE_P)
    {
    fprintf(fopen_x,"%d\n",mx);
    fprintf(fopen_y,"%d\n",my);
    }

    fclose(fopen_x);
    fclose(fopen_y);

        /-------
}
```

2:Rate control

```
int ff_mpv_encode_picture(AVCodecContext *avctx, AVPacket *pkt,
                          const AVFrame *pic_arg, int *got_packet)
{
--------
    ret = encode_picture(s, s->picture_number);

    if (gop_flag==0)  // Find the first GOP
            {
                    gop_flag=1;
                    ROP=put_bits_count(&s->pb);  //coded bits for I-frame using only the BQS--BC
                    b_qscale=s->qscale;  // BQS--BC

            }
               if (scenchange_flag!=1)  //No scen change for the first GOP
               {

               BCI= ROP;    //BCI clculation
               BCP= ROP*BP/BI;//BCo clculation
               Prate=(double)(BCI+BCP*9)*25/10; //Prate claculation
                if (s->pict_type==AV_PICTURE_TYPE_I)
                   AQS1= (Prate-Trate)*Si/128000;
               else AQS1= (Prate-Trate)*Sp/128000;

               }
-------
 if( (( (((ROP*Trate)/Prate)-put_bits_count(&s->pb))>128000*0.25 || (((ROP*Trate)/Prate)-
put_bits_count(&s->pb))<(-128000*0.25) )) &&P_count!=0 )  //scen change
```

```
            {
             printf("\n a---------------------------------- \n");
             BI=2;  //for Akiyo BI default = 5 ; for foreman BI default = 2
             BCI=ROP;
             BCP= ROP*BP/BI;
}
----------
  /*Bits count -shanglin*/
      Trate=s->bit_rate;  //Get the Trans-bits

      scenchange_flag=0;  //new gop change set the flag
      if (s->pict_type==AV_PICTURE_TYPE_I)
      {

         if (P_count!=0&&I_count!=0)
         {

         // if (scenchange_flag==0) scenchange_flag=1,

          BI=(I_total*P_count)/(P_total* I_count);  //update the BI
          BCI=ROP;  //Recalculate the BCI
          BCP=ROP*BP/BI;//Recalculate the BCP

          if (BI<1) BI=1;
          //if (BI>10) BI=10,out_bits=0,tar_bits=0;
         }
         I_count++;  //count the I-FRAME number
         I_total+=s->frame_bits;//count the I-FRAME bits
         if(flag_spatial==1) flag_spatial=0,c2=1;

      }
      else
      {

       /*
       if ((s->frame_bits-BCP)>1500||scenchange_flag==1)
       {
          BI=2;
          scenchange_flag=1;
          BCP=ROP*BP/BI;
          Prate=(BCI+BCP*9)*25/10;
       }*/
       P_total+=s->frame_bits;//count the p-FRAME number
       P_count++;//count the p-FRAME bits

      }
     printf("\nThe BI=  %d \n",BI);


         BCI=ROP;//Recalculate the BCI
         BCP=ROP*BP/BI;//Recalculate the BCP
         Prate=(double)(BCI+BCP*9)*25/10;//Recalculate the Prate
```

```
        printf("\n Prate %f ROP %ld b->qscale %d \n",Prate,ROP,b_qscale);


       AQS1= (Prate-Trate)/128000; ////Recalculate the AQS1


      printf(" the bits of the AQS1 = %f\n ", AQS1);

      if (s->pict_type==AV_PICTURE_TYPE_I) T_d=(BCI*Trate)/Prate;  //the desire T-bits
      else T_d=(BCP*Trate)/Prate;


      out_bits+=(double)(s->frame_bits-Trate/25);
      tar_bits+=(double)(T_d-Trate/25);
      printf(" the s->frame_bits = %d\n ", s->frame_bits);
      printf(" the out_bits = %f\n ", out_bits);
      printf(" the tar_bits = %f\n ", tar_bits);


      AQS2=(out_bits-tar_bits)/128000;////Recalculate the AQS2



      printf(" the bits of the AQS2 = %f\n ", AQS2);
      printf("the scen_cou = %d \n",scen_cou);
      -----
}


static av_always_inline void encode_mb_internal(MpegEncContext *s,
                                                int motion_x, int motion_y,
                                                int mb_block_height,
                                                int mb_block_width,
                                                int mb_block_count)
{
-------

       if (!(s->mpv_flags & FF_MPV_FLAG_QP_RD)) {
          s->qscale = s->current_picture_ptr->qscale_table[mb_xy];
          s->dquant = s->qscale - last_qp;

          if (s->out_format == FMT_H263) {
             s->dquant = av_clip(s->dquant, -2, 2);

             if (s->codec_id == AV_CODEC_ID_MPEG4) {
                if (!s->mb_intra) {
                   if (s->pict_type == AV_PICTURE_TYPE_B) {
                      if (s->dquant & 1 || s->mv_dir & MV_DIRECT)
                         s->dquant = 0;
                   }
                   if (s->mv_type == MV_TYPE_8X8)
                      s->dquant = 0;
```

```
                }
            }
        }
    }
    ff_set_qscale(s, last_qp + s->dquant);
  } else if (s->mpv_flags & FF_MPV_FLAG_QP_RD)
  {

    ff_set_qscale(s, s->qscale + s->dquant);
  }
 if (ROP!=0)  //set my qscale
  {


  c1=(double)128000/(s->bit_rate);
  c1=c1*1.5;

  if (AQS2<2&&AQS2>0) AQS2=AQS2*4;

  if(s->pict_type==AV_PICTURE_TYPE_I)  //I/P frame
  my_qscale=b_qscale+(c1*AQS1+2*AQS2*c2)*Si;
  else my_qscale=b_qscale+(c1*AQS1+2*AQS2*c2)*Sp;
  s->qscale=my_qscale;
  ff_set_qscale(s, s->qscale); //set qscale
  }

-------
}
```