Checkmate!

Paul Ardis
Tom Scogland
Andy Zeigler
Overview

1) Introduction
2) Goals
3) Prior Work
4) Implementation
5) Challenges
Overview

1) Introduction
2) Goals
3) Prior Work
4) Implementation
5) Challenges
Introduction

• Design an autonomous and portable mixed-reality system that enables a computer to play games against a human
• Will use standard chessboard and pieces
Overview

1) Introduction
2) Goals
3) Prior Work
4) Challenges
5) Implementation
Goals

• Goals
  – Design recognition engine
  – Focus our efforts on recognition and arbitration
  – Build simple AI
Overview

1) Introduction
2) Goals
3) Prior Work
4) Implementation
5) Challenges
Prior Work

• 2002, Ecole Polytechnique
  – 3 students design autonomous chess playing robot with vision and a robotic arm

• Jamca
  – A small, simple, open-source chess board implementation
Prior Work

• Internet chess community
  – Many, many playing engines and board display programs
  – Most are open and/or interoperable via existing standards
  – We aim for standards compliance!!
Overview

1) Introduction
2) Goals
3) Prior Work
4) Implementation
   a) Representation
   b) Calibration
   c) Gameplay
5) Challenges
Representation

• Checkmate “knows” how to play chess
• Board is represented in internal structures
• Implements all chess rules
  – 50-move rule
  – En-passant
  – Castling
The Board

<table>
<thead>
<tr>
<th>110</th>
<th>111</th>
<th>112</th>
<th>113</th>
<th>114</th>
<th>115</th>
<th>116</th>
<th>117</th>
<th>118</th>
<th>119</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>101</td>
<td>102</td>
<td>103</td>
<td>104</td>
<td>105</td>
<td>106</td>
<td>107</td>
<td>108</td>
<td>109</td>
</tr>
<tr>
<td>90</td>
<td>91</td>
<td>92</td>
<td>93</td>
<td>94</td>
<td>95</td>
<td>96</td>
<td>97</td>
<td>98</td>
<td>99</td>
</tr>
<tr>
<td>80</td>
<td>81</td>
<td>82</td>
<td>83</td>
<td>84</td>
<td>85</td>
<td>86</td>
<td>87</td>
<td>88</td>
<td>89</td>
</tr>
<tr>
<td>70</td>
<td>71</td>
<td>72</td>
<td>73</td>
<td>74</td>
<td>75</td>
<td>76</td>
<td>77</td>
<td>78</td>
<td>79</td>
</tr>
<tr>
<td>60</td>
<td>61</td>
<td>62</td>
<td>63</td>
<td>64</td>
<td>65</td>
<td>66</td>
<td>67</td>
<td>68</td>
<td>69</td>
</tr>
<tr>
<td>50</td>
<td>51</td>
<td>52</td>
<td>53</td>
<td>54</td>
<td>55</td>
<td>56</td>
<td>57</td>
<td>58</td>
<td>59</td>
</tr>
<tr>
<td>40</td>
<td>41</td>
<td>42</td>
<td>43</td>
<td>44</td>
<td>45</td>
<td>46</td>
<td>47</td>
<td>48</td>
<td>49</td>
</tr>
<tr>
<td>30</td>
<td>31</td>
<td>32</td>
<td>33</td>
<td>34</td>
<td>35</td>
<td>36</td>
<td>37</td>
<td>38</td>
<td>39</td>
</tr>
<tr>
<td>20</td>
<td>21</td>
<td>22</td>
<td>23</td>
<td>24</td>
<td>25</td>
<td>26</td>
<td>27</td>
<td>28</td>
<td>29</td>
</tr>
<tr>
<td>10</td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
<td>16</td>
<td>17</td>
<td>18</td>
<td>19</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
</tr>
</tbody>
</table>
Calibration

• How does it “see” the chess board?
  – Auto-calibrate camera for specific lighting conditions
    • Gamma, color balance, etc.
  – Calibrate board
    • User clicks on corners
  – Checkmate knows board geometry
    • Using corners of board, it unprojects board plane to an orthogonal view
Gameplay

- Stores camera shot
- Calculates perspective matrix
- Divides unit square into regions
- Maps camera regions to unit regions
- Determines changes to centers of squares
- Finds most-changed
Gameplay (cont.)

Move Finding

Unit Square

Actual Image
Gameplay (cont.)

- Artificial Intelligence
  - Input: List of legal moves
  - Output: A “good” move
  - Point system (10, 5, 3, 3, 1)
Gameplay cont.

• Artificial Intelligence (cont.)
  – Classification of moves
    • Capture
    • Risk
    • Evasion
  – Want to capture safely if possible
Implementation

• Demo time
  – Play a game of chess
Overview

1) Introduction
2) Goals
3) Prior Work
4) Implementation
5) Challenges
Challenges

- Accurately handling real-world piece movement
  - Black VS white pieces
  - Nonstandard board/piece colors
  - Abnormal lighting, off-color or off-center
  - Getting a reliable camera angle to avoid occlusion
Future Work

• Smooth and natural autonomous interaction
  – Showing which piece the computer wants to move, and to where
  – Explaining what is wrong with an illegal move that has been made by the player
Discussion

• Q/A